Reachability Problems for Hybrid Automata

J-F Raskin
Université Libre de Bruxelles
based on joint works with
T. Brihaye, L. Doyen, G. Geeraerts, T. Henzinger, J. Ouaknine, J. Worrell

Content

- Motivations: reactive embedded and hybrid systems
- **Classes** of hybrid automata
- **Symbolic semi-**algorithm for reachability
- Reachability problem: **decidability** frontier
- **Approximate** reachability
- **Time-bounded** reachability

Reactive and hybrid systems

Reactive systems maintain a continuous interaction with their environment

- non-terminating
- respect/enforce real-time properties
- cope with concurrency
- embedded in complex-continuous-critical env

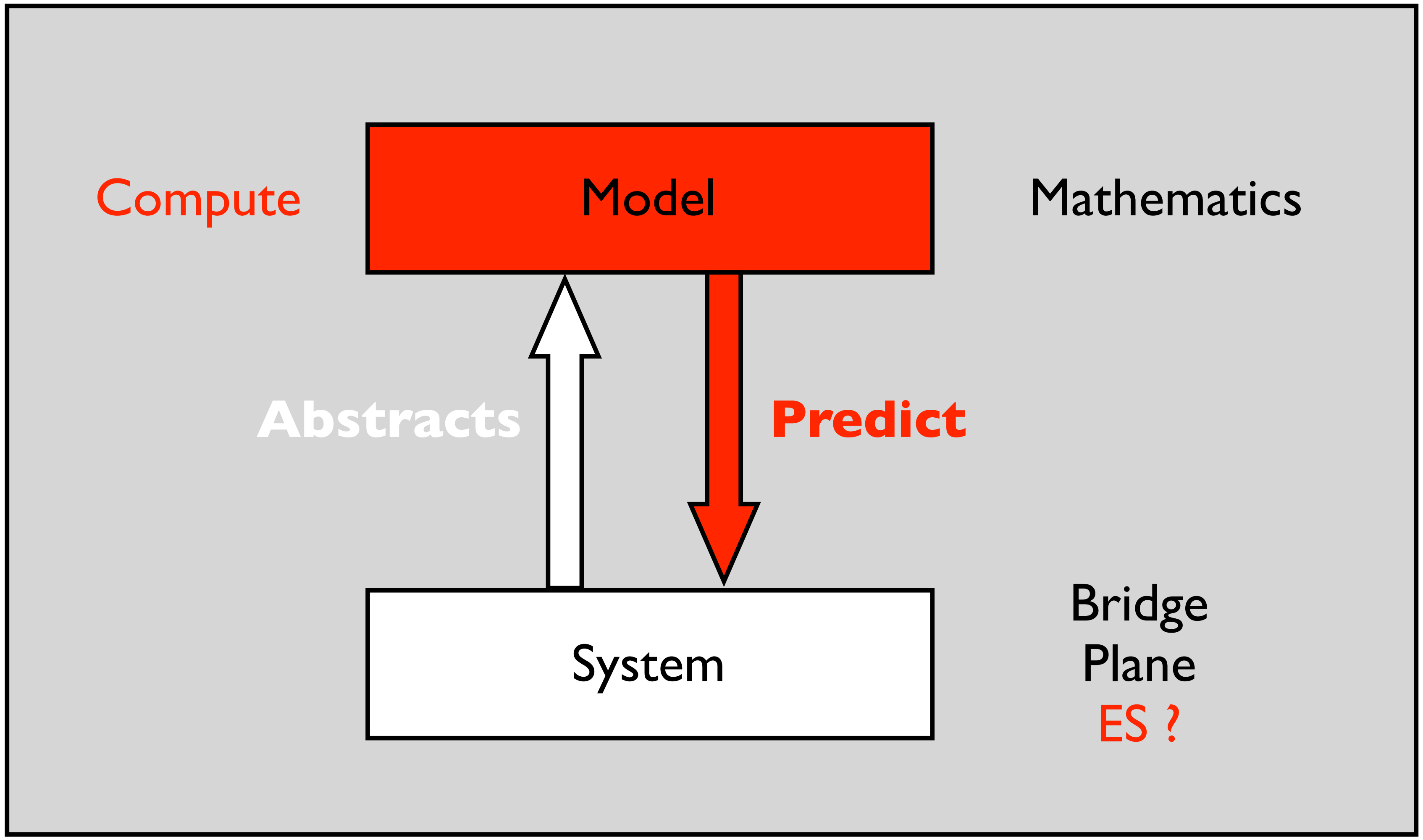
→ difficult to develop correctly

300 horses power

100 processors

Is the software correct ?

How to cope with complexity



Hybrid automata

Mixing discrete-continuous evolutions

- Finite state automata to model (discrete) reactive systems
- Differential equations to model continuous environments
- Hybrid automata: combine the two
  - finite automata + continuous variables
  - discrete transitions + differential equations

Example

- Three environment components:

  - A tank containing water;
  - A gas burner that can be turn on or off;
  - A digital thermometer that monitors the temperature within the tank.

  and a controller

- We want to design a controller strategy that maintains the temperature within an interval of safe temperatures.

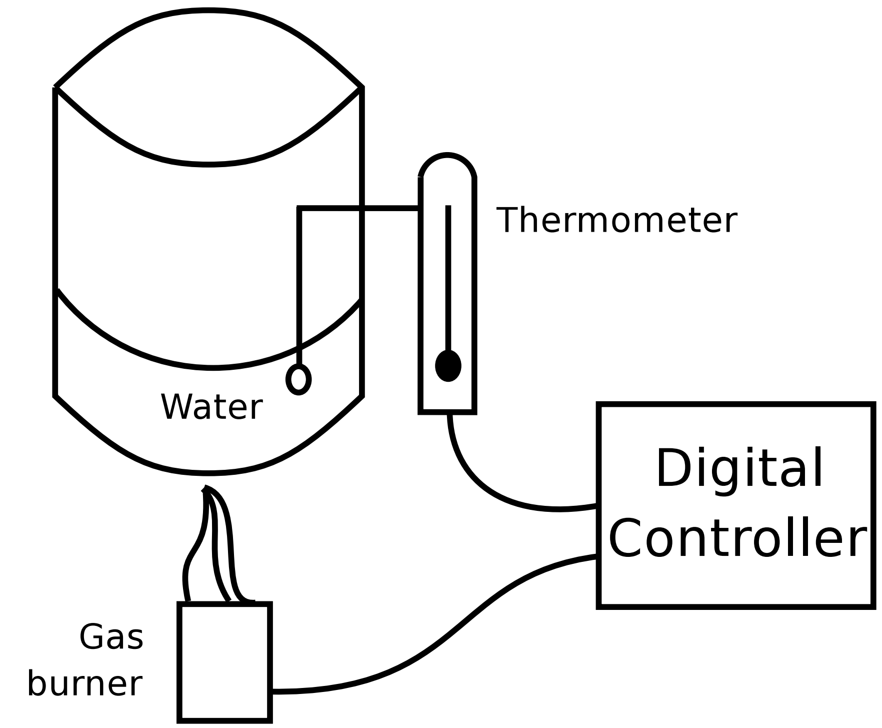


Fig. 1. Our running example

Continuous part

- Behavior of the temperature in the tank

  -Mode **OFF**: $\mathbf{x(t) = I\, e^{-Kt}}$, i.e. $\dot{x} = -Kx$

  -Mode **ON**: $\mathbf{x(t) = I\, e^{-Kt} + h\,(1 - e^{-Kt})}$, i.e. $\dot{x} = K(h-x)$

  I=initial temperature of the water
  K=constant (nature of the tank)
  h=constant (power gas burner)
  t=time.

- **ON** and **OFF**=modes of the tank evolution

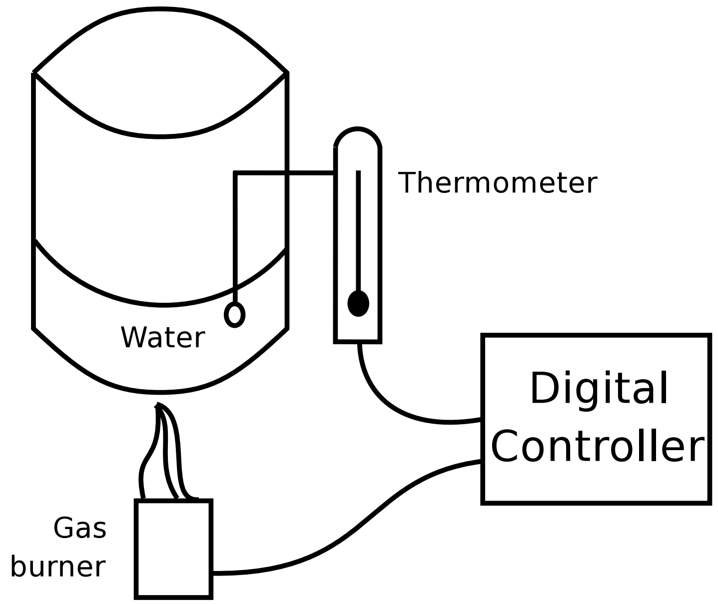


Fig. 1. Our running example

Evolution of the temperature

Mode changes

Continuous Evolutions

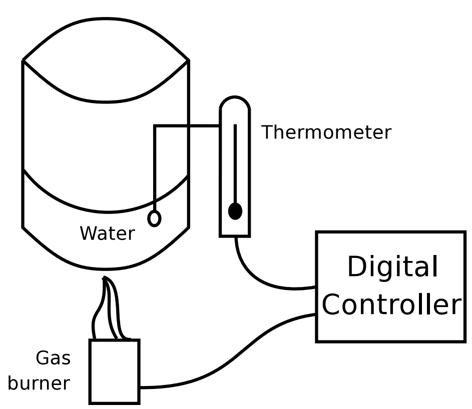


Fig. 1. Our running example

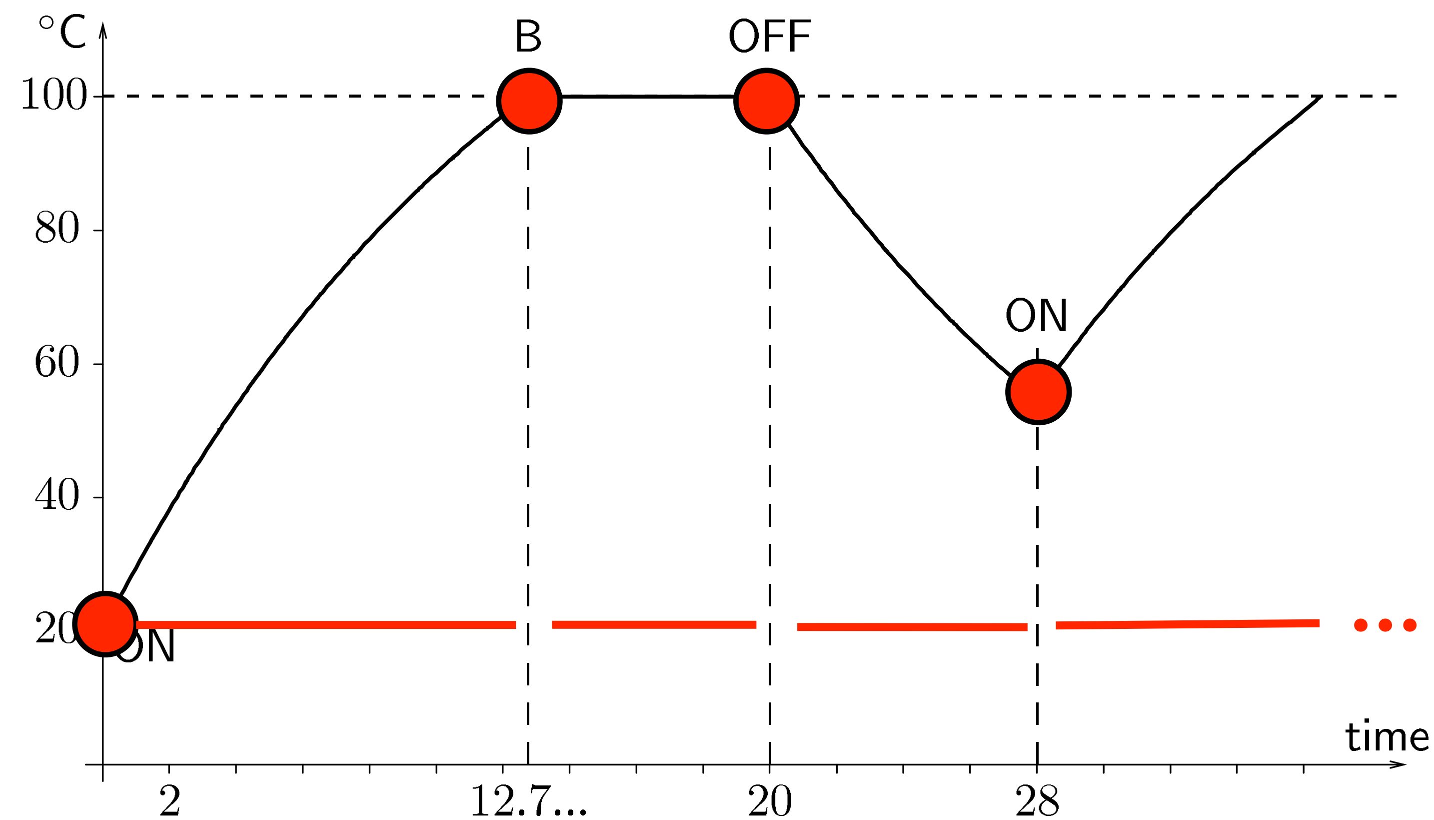
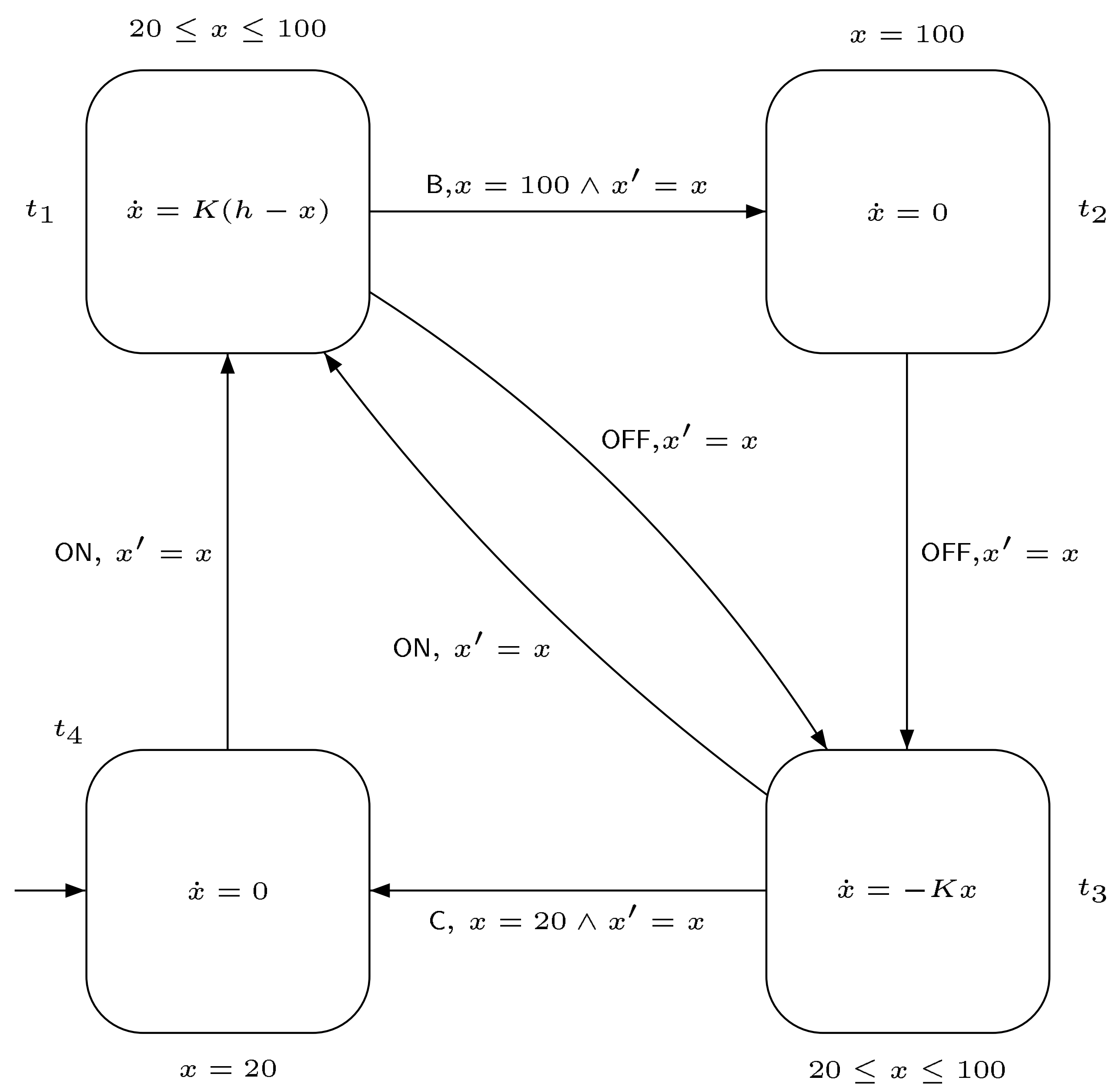


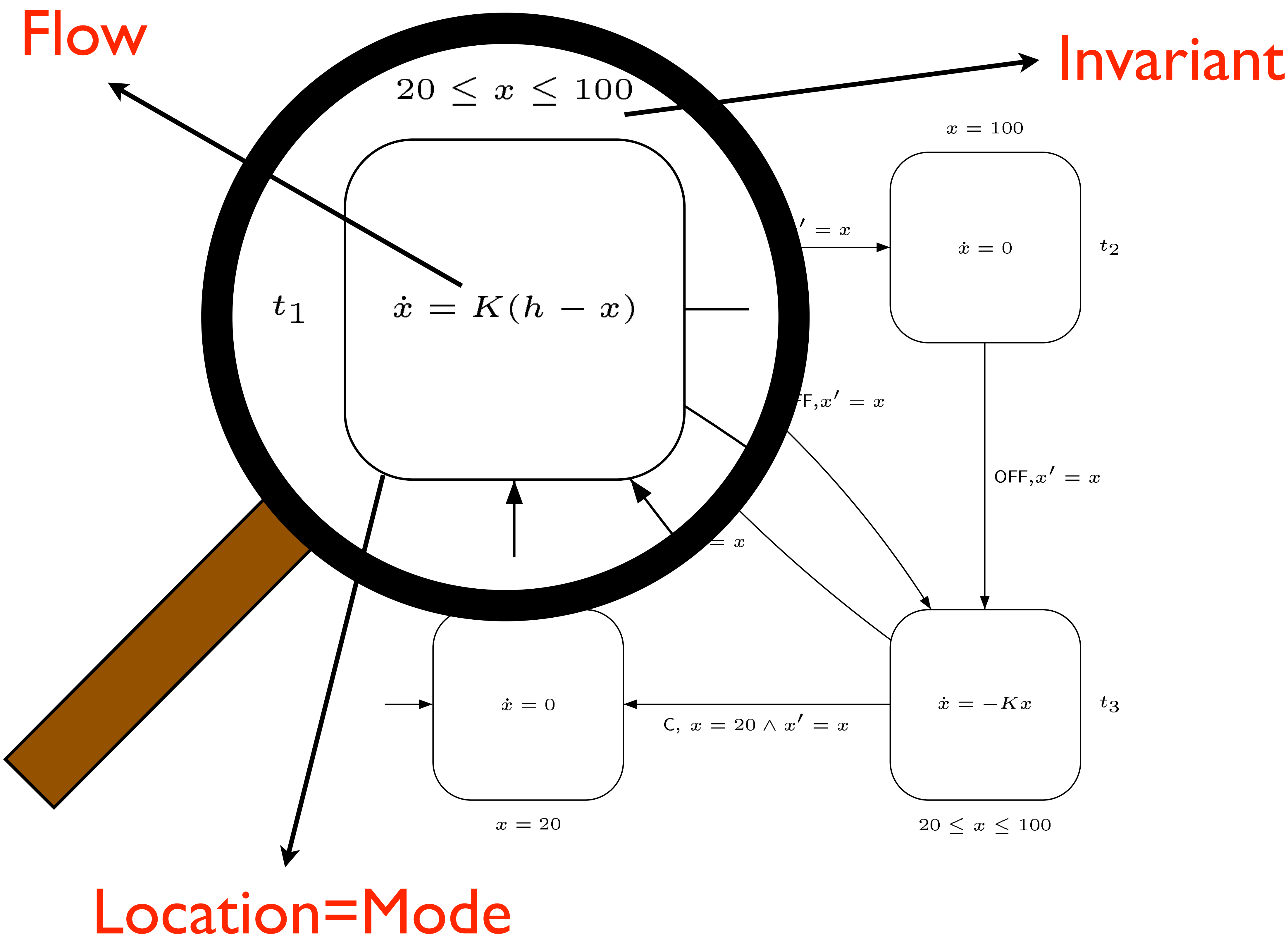
Fig. 2. One possible behavior of the tank

Evolution of the temp. is **not** purely continuous. It depends on the mode **ON** and **OFF** for example, and that it is below 100° or not.

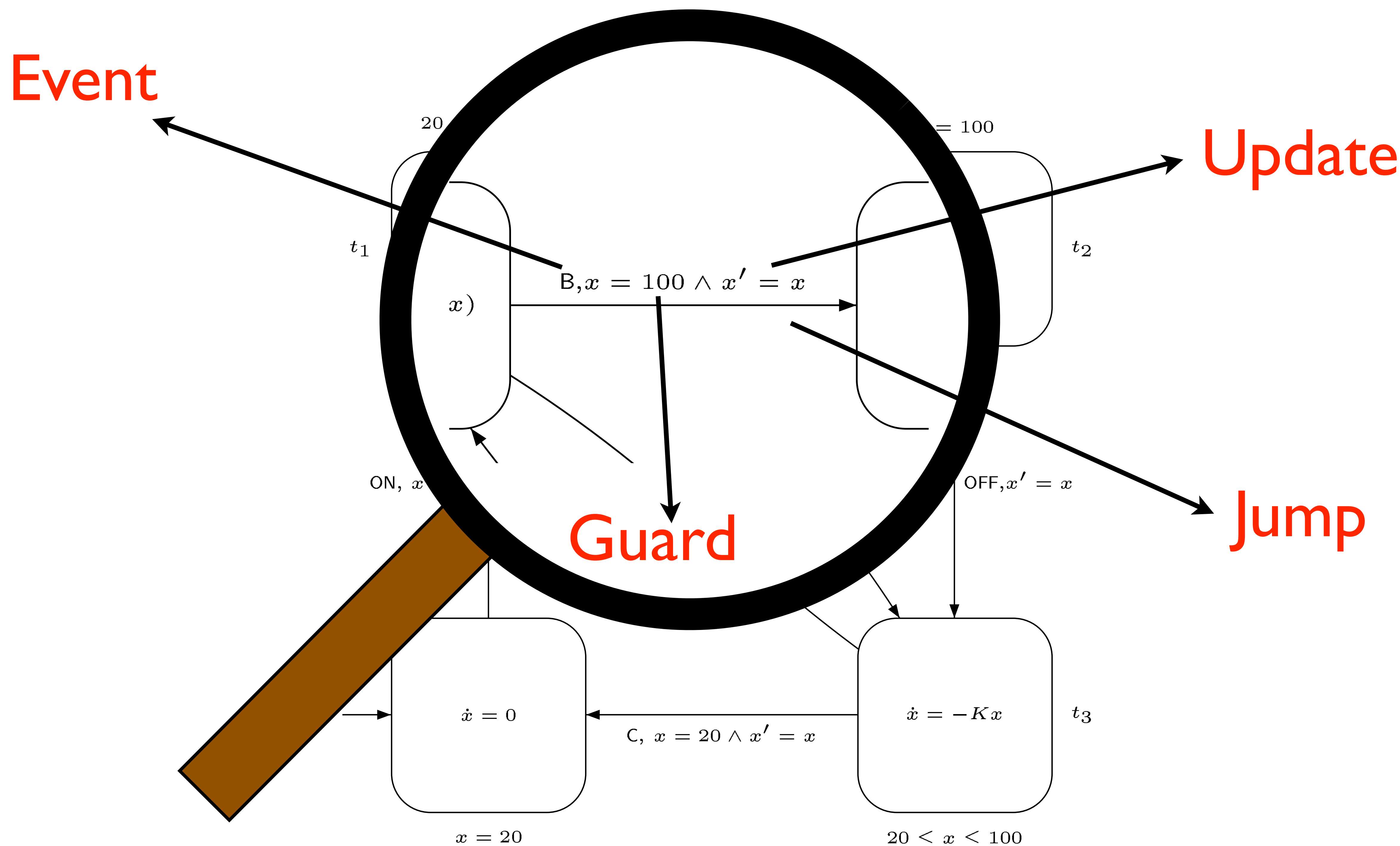
An HA for the tank



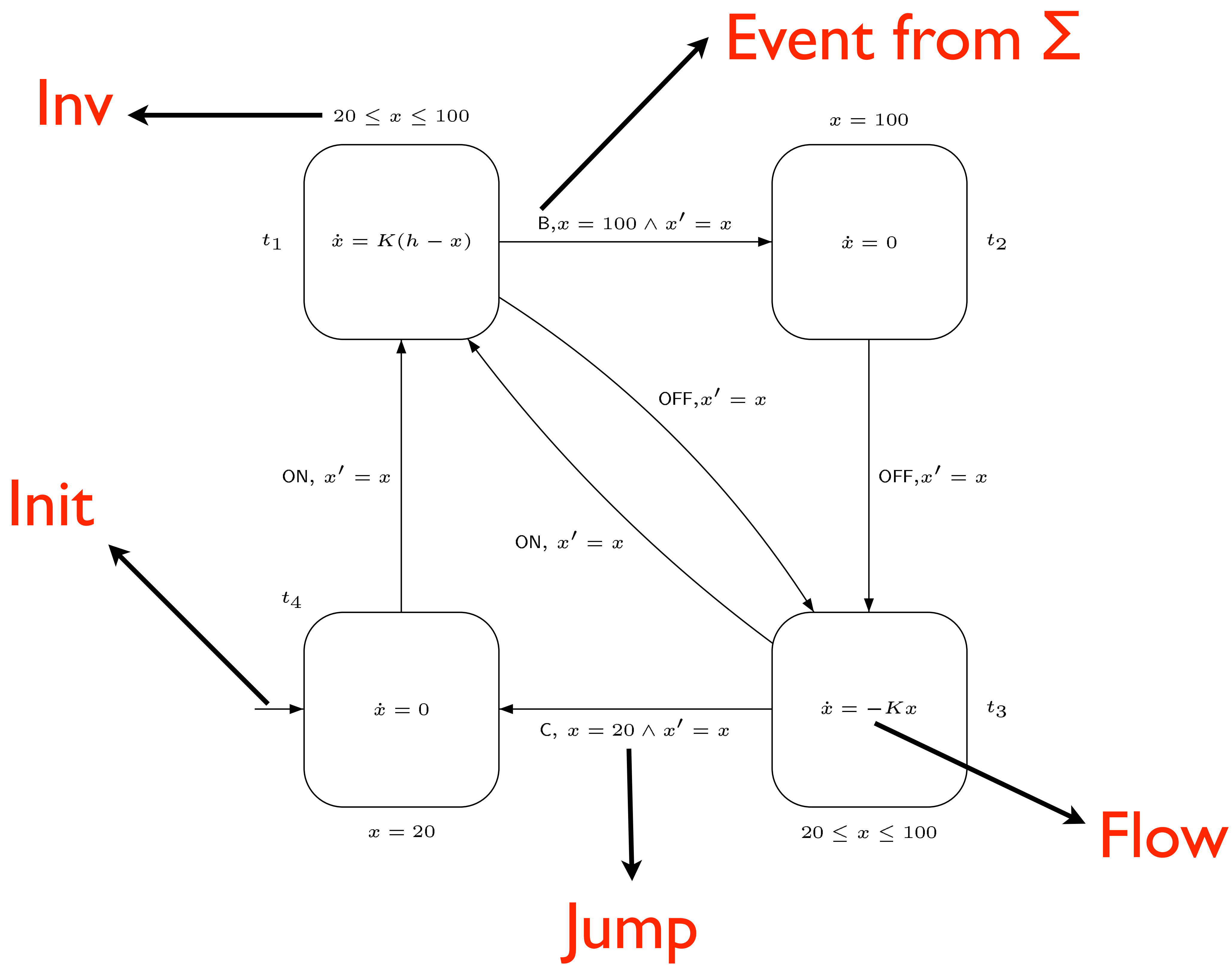
An HA for the tank



An HA for the tank



An HA for the tank



Hybrid automata - Syntax

Definition

H=(Loc,Σ,Edge,X,Init,Inv,Flow,Jump), where:

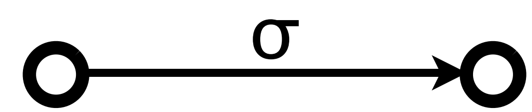
- **Loc** is a finite set $\{l_1,l_2,\ldots,l_n\}$ of (control locations) modeling control modes
- **Σ** is a finite set of event names
- **Edge** ⊆ Loc × Σ × Loc is a finite set of labelled edges modeling discrete changes between control modes
- **X** is a finite set $\{x_1,x_2,\ldots,x_m\}$ of real-valued variables.
  - We write $\dot{X}=\{\dot{x}_1,\dot{x}_2,\ldots,\dot{x}_m\}$ for the dotted variables and
  - $X'=\{x'_1,x'_2,\ldots,x'_m\}$ for the primed variables
- **Init(X)**, **Inv(X)**, and **Flow(X,$\dot{X}$)** are predicates associated to locations
- **Jump(X,X')** is a function that assigns a predicate to each labelled edge

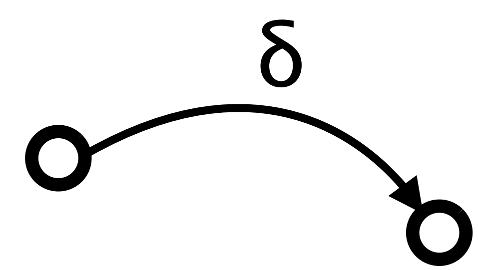
TTS of a HA

- Let H=(Loc,Σ,Edge,X,Init,Inv,Flow,Jump) be a HA.
- Its associated **T**imed **T**ransition **S**ystem ⟦H⟧=(S,S_0,Σ,→) is defined as follows:
  - S is the set of pairs (l,v) where l∈Loc and v∈⟦Inv(l)⟧;
  - S_0 is the subset of pairs (l,v)∈S such that v∈⟦Init(l)⟧;

Timed transition system of a HA

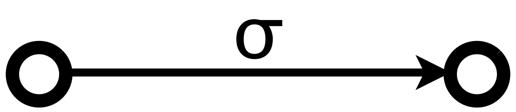
Transition relation



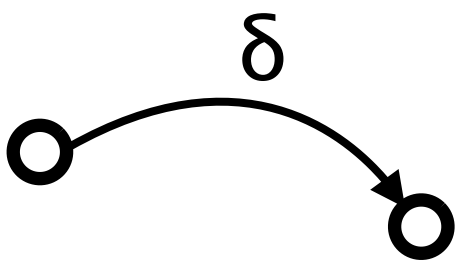


Timed transition system of a HA

Transition relation

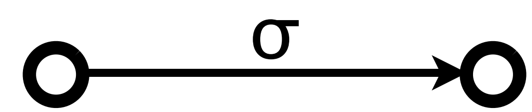


- discrete steps:
  for each edge $e=(l,\sigma,l')\in E$, we have $(l,v)\rightarrow_{\sigma}(l',v')$
  if $(l,v)\in S$, $(l',v')\in S$ and $(v,v')\in[\![Jump(e)]\!]$;

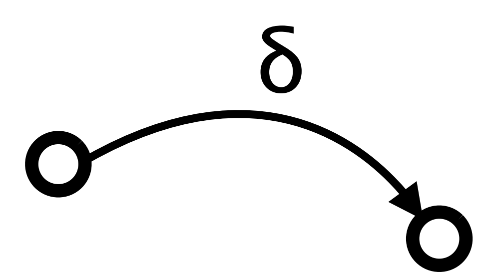


Timed transition system of a HA

Transition relation

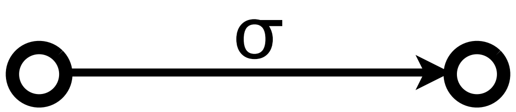


- discrete steps:
  for each edge $e=(l,\sigma,l')\in E$, we have $(l,v)\rightarrow_{\sigma}(l',v')$
  if $(l,v)\in S$, $(l',v')\in S$ and $(v,v')\in[\![Jump(e)]\!]$;
- continuous steps: for each $\delta\in\mathbb{R}\geq 0$, we have $(l,v)\rightarrow_{\delta}(l',v')$
  if $(l,v)\in S$, $(l',v')\in S$, $l=l'$,
  and there exists a *differentiable function* $f:[0,\delta]\rightarrow\mathbb{R}^m$,
  with derivative $\dot{f}(0,\delta)\rightarrow\mathbb{R}^m$
  such that :
  1) $f(0)=v$,
  2) $f(\delta)=v'$ and
  3)for all $\varepsilon\in(0,\delta)$, both



Timed transition system of a HA

Transition relation



- discrete steps:

  for each edge $e=(l,\sigma,l')\in E$, we have $(l,v)\rightarrow_{\sigma}(l',v')$

  if $(l,v)\in S$, $(l',v')\in S$ and $(v,v')\in[\![Jump(e)]\!]$;

- continuous steps: for each $\delta\in\mathbb{R}_{\geq 0}$, we have $(l,v)\rightarrow_{\delta}(l',v')$

  if $(l,v)\in S$, $(l',v')\in S$, $l=l'$,

  and there exists a *differentiable function* $f:[0,\delta]\rightarrow\mathbb{R}^m$,

  with derivative $\dot{f}(0,\delta)\rightarrow\mathbb{R}^m$

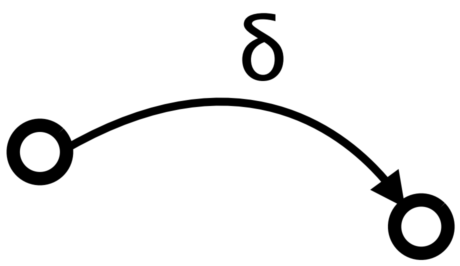
  such that :

  1) $f(0)=v$,

  2) $f(\delta)=v'$ and

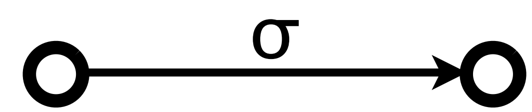
  3)for all $\varepsilon\in(0,\delta)$, both

  - $f(\varepsilon)\in[\![Inv(l)]\!]$ and



Timed transition system of a HA

Transition relation



- discrete steps:

  for each edge $e=(l,\sigma,l')\in E$, we have $(l,v)\rightarrow_{\sigma}(l',v')$

  if $(l,v)\in S$, $(l',v')\in S$ and $(v,v')\in[\![Jump(e)]\!]$;

- continuous steps: for each $\delta\in\mathbb{R}_{\geq 0}$, we have $(l,v)\rightarrow_{\delta}(l',v')$

  if $(l,v)\in S$, $(l',v')\in S$, $l=l'$,

  and there exists a *differentiable function* $f:[0,\delta]\rightarrow\mathbb{R}^m$,

  with derivative $\dot{f}(0,\delta)\rightarrow\mathbb{R}^m$

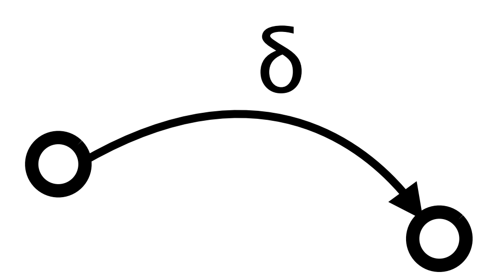
  such that :

  1) $f(0)=v$,

  2) $f(\delta)=v'$ and

  3)for all $\varepsilon\in(0,\delta)$, both

  - $f(\varepsilon)\in[\![Inv(l)]\!]$ and
  - $(f(\varepsilon),\dot{f}(\varepsilon))\in[\![Flow(l)]\!]$.



Reachability

- Let $Path_F(S_0)$=set of finite paths starting from a state in S_0
- Let $T=(S,S_0,\Sigma,\rightarrow)$ be a TTS
  Let $\lambda=s_0\tau_0s_1\tau_1\ldots s_n \in Path_F(T)$
  State(λ) denotes the set of states that appear along λ
- We say that a path λ reaches a state s if $s \in State(\lambda)$
- We say that s is reachable in T if $s \in \cup_{\lambda \in PathF(T)} State(\lambda)$
- Reach(T) denotes the set of states reachable in T

Safety and reachability

- A set of state $R \subseteq S$ is called a region.
- A region R is reachable in T iff $R \cap Reach(T) \neq \emptyset$.
- The rechability problem associated to a TTS T and a region R asks if $R \cap Reach(T) \neq \emptyset$.
- The safety problem associated to a TTS T and a region R asks if $Reach(T) \subseteq R$.
- Those two problems are dual in the following formal sense:

  Let R be a region and $R' = S \setminus R$.

$Reach(T) \subseteq R$ iff $R' \cap Reach(T) = \emptyset$.

Classes of Hybrid Automata

Classes of HA

Linear HA

- **Linear** flow constraints:
  Lin($X^\bullet$), **ex:** $x^\bullet = y^\bullet + 3$

- **Linear** guards and updates:
  Lin(X)→Lin(X,X'),
  ex: $x+y<1 \rightarrow x' := y+2$

Classes of HA

Linear HA

- **Linear** flow constraints:
  Lin($X^\bullet$), **ex:** $x^\bullet=y^\bullet+3$
- **Linear** guards and updates:
  Lin(X)→Lin(X,X'),
  ex: $x+y<1 \rightarrow x':=y+2$

Rectangular HA

- **Rectangular** flow constraints:
  Rect($X^\bullet$), **ex**: $x^\bullet\in[1,2]\wedge y^\bullet\in[2,5]$
- **Rectangular** guards-updates:
  Rect(X)→Rect(X')
  ex: $x\in[2,5]\rightarrow x'\in[5,7]$

Classes of HA

Linear HA

- **Linear** flow constraints: Lin($X^\bullet$), **ex:** $x^\bullet = y^\bullet + 3$
- **Linear** guards and updates: Lin(X)→Lin(X,X'), **ex:** $x+y<1 \rightarrow x' := y+2$

Affine HA

- **Affine** flow constraints: Aff($X, X^\bullet$), ex: $x^\bullet = 2x+3y$
- **Linear** guards and updates: Lin(X)→Lin(X,X'), **ex:** $x+y<1 \rightarrow x' := y+2$

Rectangular HA

- **Rectangular** flow constraints: Rect($X^\bullet$), **ex**: $x^\bullet \in [1,2] \wedge y^\bullet \in [2,5]$
- **Rectangular** guards-updates: Rect(X)→Rect(X') **ex**: $x \in [2,5] \rightarrow x' \in [5,7]$

Classes of HA

Linear HA

- **Linear** flow constraints: $Lin(X^{\bullet})$, **ex:** $x^{\bullet}=y^{\bullet}+3$
- **Linear** guards and updates: $Lin(X) \rightarrow Lin(X,X')$, **ex:** $x+y<1 \rightarrow x':=y+2$

Affine HA

- **Affine** flow constraints: $Aff(X,X^{\bullet})$, ex: $x^{\bullet}=2x+3y$
- **Linear** guards and updates: $Lin(X) \rightarrow Lin(X,X')$, **ex:** $x+y<1 \rightarrow x':=y+2$

Rectangular HA

- **Rectangular** flow constraints: $Rect(X^{\bullet})$, **ex**: $x^{\bullet} \in [1,2] \wedge y^{\bullet} \in [2,5]$
- **Rectangular** guards-updates: $Rect(X) \rightarrow Rect(X')$ **ex**: $x \in [2,5] \rightarrow x' \in [5,7]$

O-minimal HA

- Use of **O-minimal theory**
- **Strong resets**: all variables are reset during any mode change

Symbolic Semi-Algorithm for RHA/LHA

Effective procedure for Post in RHA

Effective procedure for Post in RHA

- A linear term over X is a linear combination of the variables in X with integer coefficients.

  ex : $3x+2y-1$.

- A linear formula over X is a boolean combination of inequalities between linear terms over X.

  ex : $3x+2y-1 \geq 0 \wedge y \geq 5$.

- Given a linear formula ψ, we write $[\![\psi]\!]$ for the set of valuations v such that $v \models \psi$.

Effective procedure for Post in RHA

- Linear formulas + quantifiers
  =T($\mathbb{R}$,0,1,+,≤).
  =The theory of reals with addition.

  This theory allows for quantifier elimination.

  ex : "$\forall y \bullet y \geq 5 \rightarrow x+y \geq 7$" is equivalent to "$x \geq 2$".

- A symbolic region of H is a finite set

$$\{ (l,\psi_l) \mid l \in Loc \} \text{ where } [\![\psi_l]\!] \subseteq [\![Inv(l)]\!].$$

Effective procedure for Post in RHA

Given a location $l \in Loc$ and a set of valuations $V \subseteq [X \rightarrow \mathbb{R}]$ such that $V \subseteq Inv(l)$, the forward time closure, noted $\langle V \rangle_l^{\nearrow}$ is the set of valuations that are reachable from some valuation $v \in V$ by letting time pass.

This set is defined as follows:

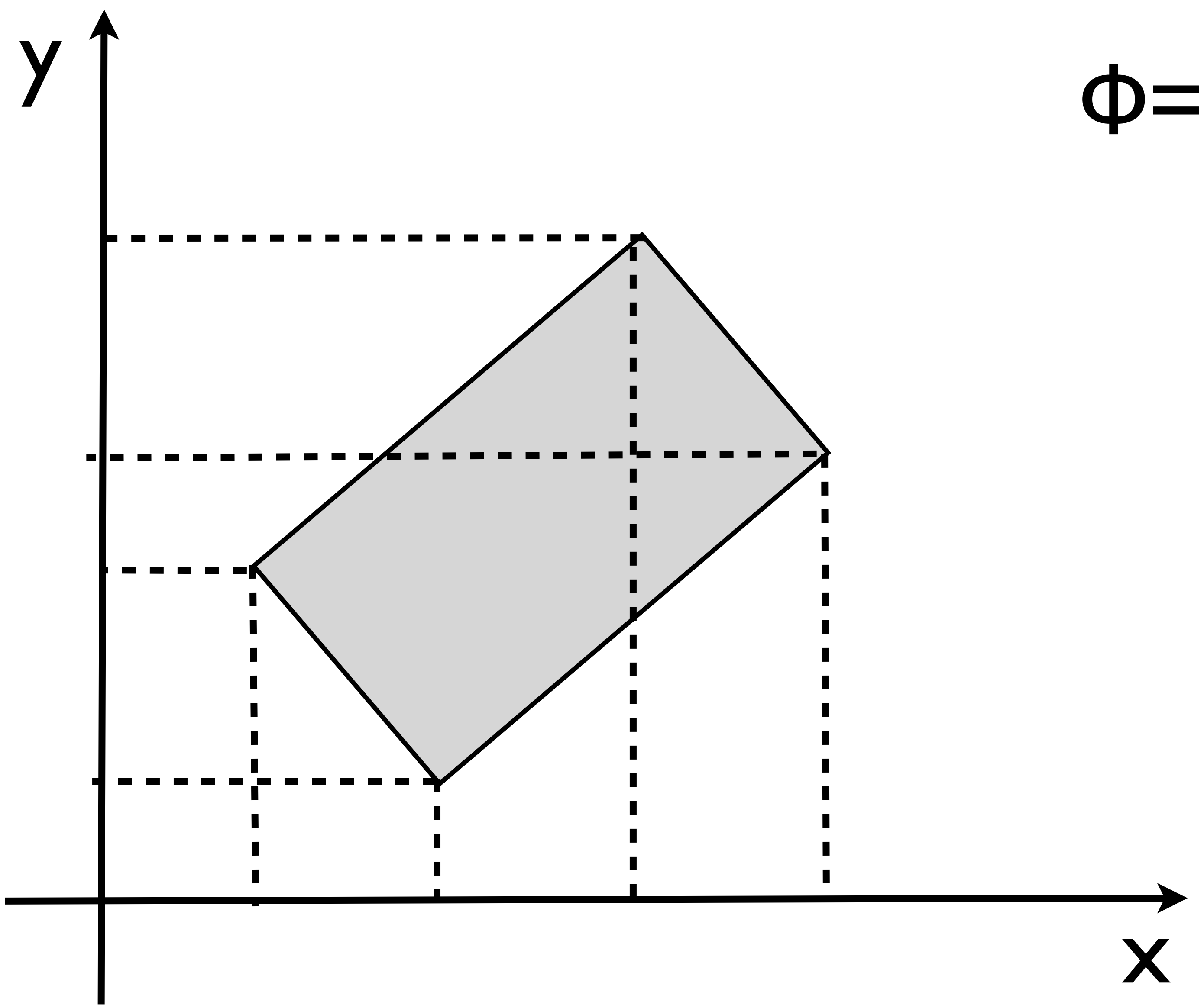
$\langle V \rangle_l^{\nearrow}$ is the set of valuation $v' \in [X \rightarrow \mathbb{R}]$ such that

$$
\begin{aligned}
&\exists \mathbf{v} \in V \bullet \exists t \in \mathbb{R}_{\geq 0} \bullet \forall x \in X \bullet \\
&\qquad \mathbf{v}(x) + t \times \mathbf{Inf}([\![Flow(l)]\!](x)) \leq v'(x) \leq \mathbf{v}(x) + t \times \mathbf{Sup}([\![Flow(l)]\!](x)) \\
&\qquad \wedge v'(x) \in [\![Inv(l)]\!].
\end{aligned}
$$

After **quantifier eliminations**, we get a boolean combination of linear constraints.

An example of time elapsing

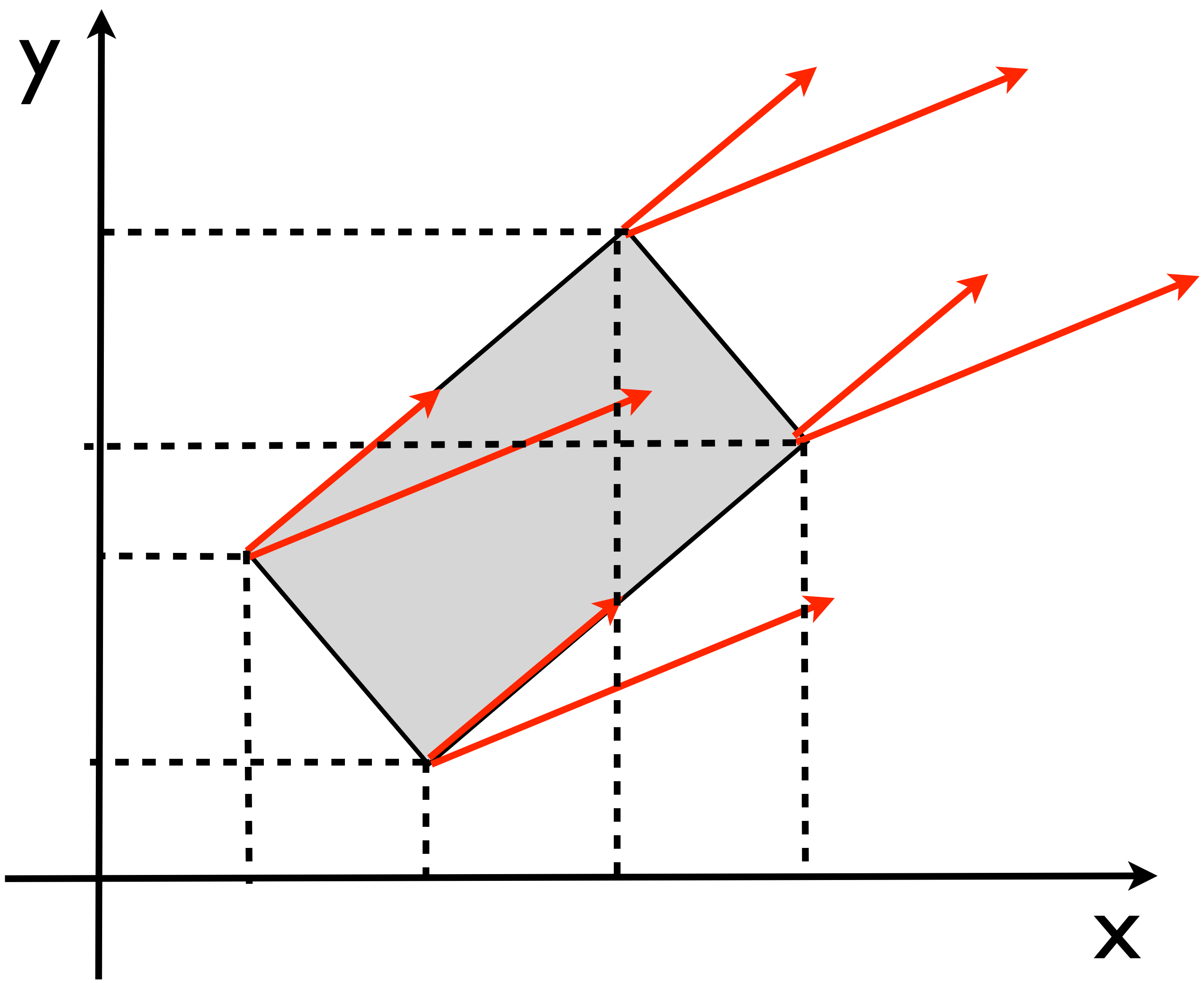
Assume $\dot{x}=[1,2]$ and $\dot{y}=1$

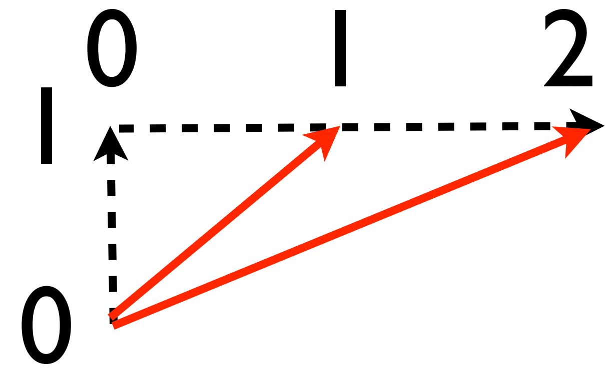


$$\Phi=\{ (x,y) \mid \quad x\in[1,4] \wedge y\in[1,6] \wedge y\geq -2x+5 \wedge \ldots \}$$

An example of time elapsing

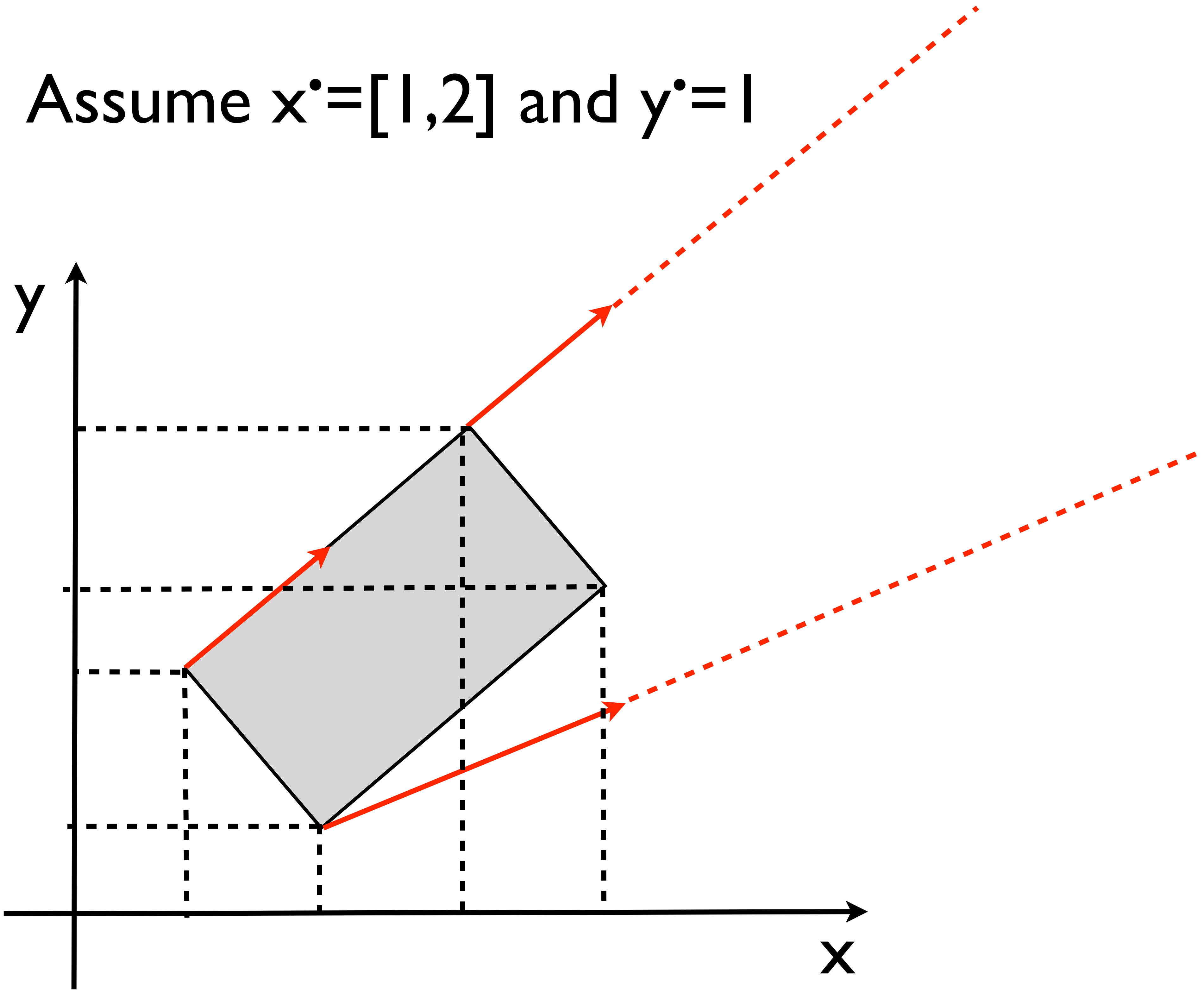
Assume $x^{\bullet}=[1,2]$ and $y^{\bullet}=1$





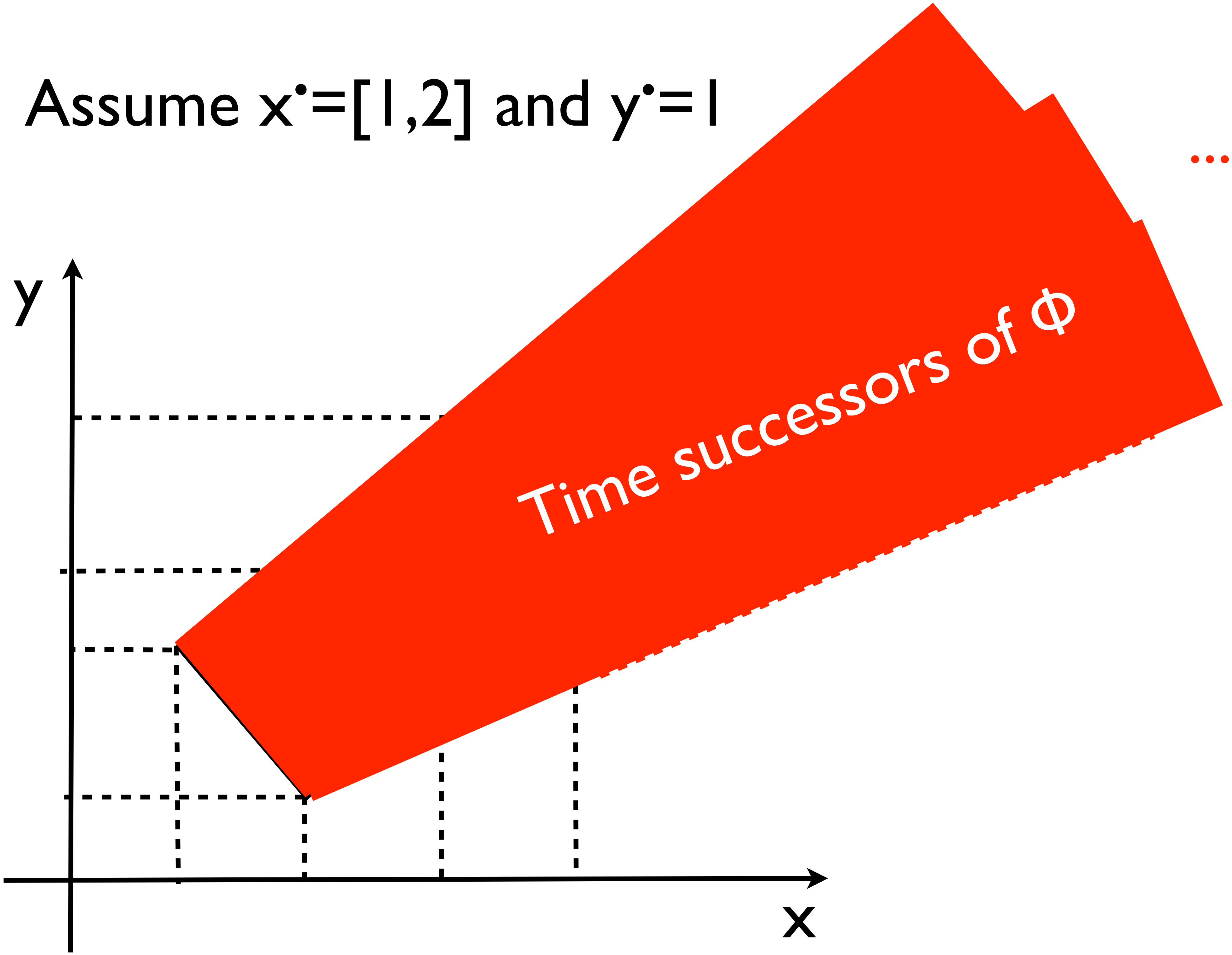
An example of time elapsing

Assume $x^{\bullet}=[1,2]$ and $y^{\bullet}=1$



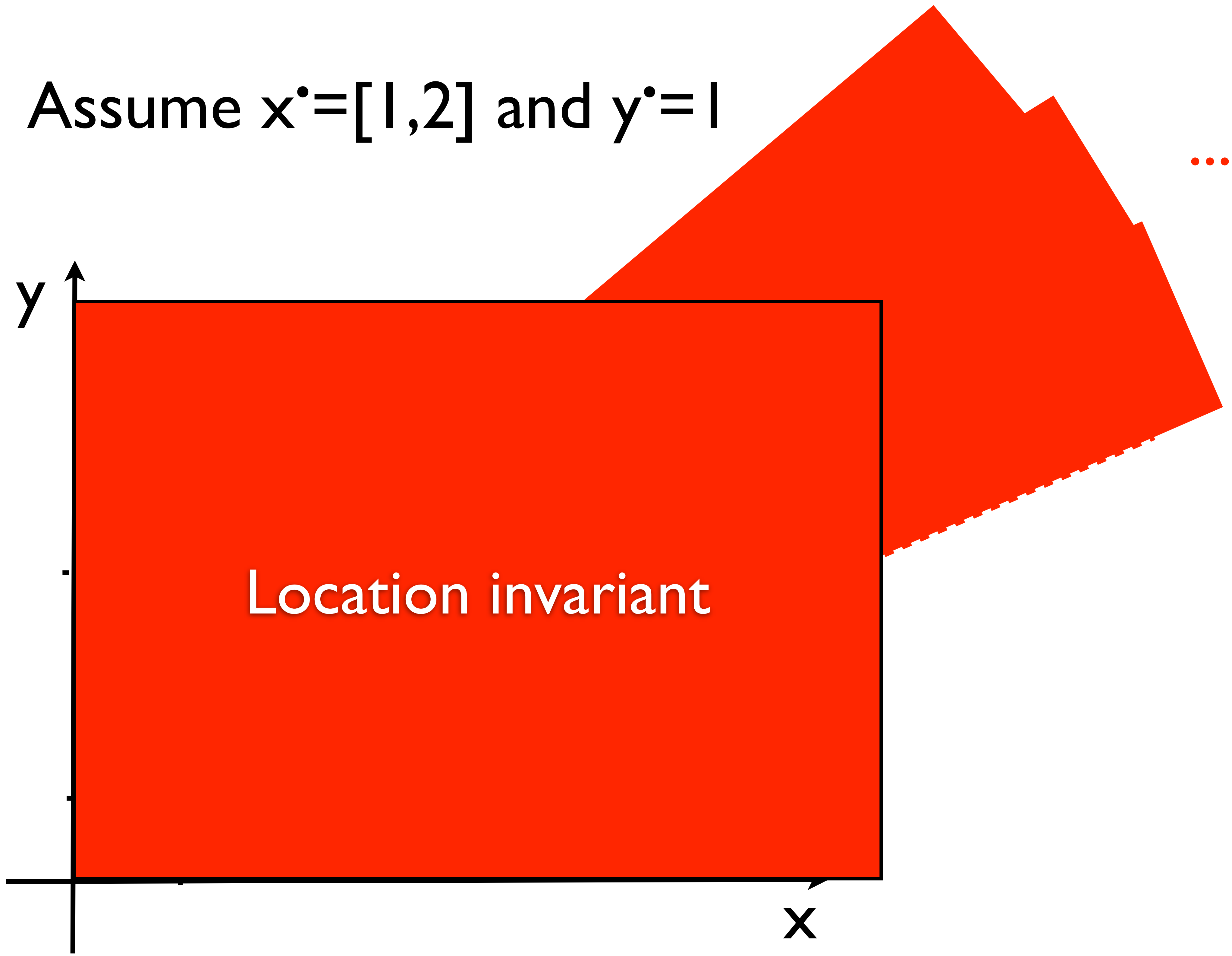
An example of time elapsing

Assume $x^{\bullet}=[1,2]$ and $y^{\bullet}=1$



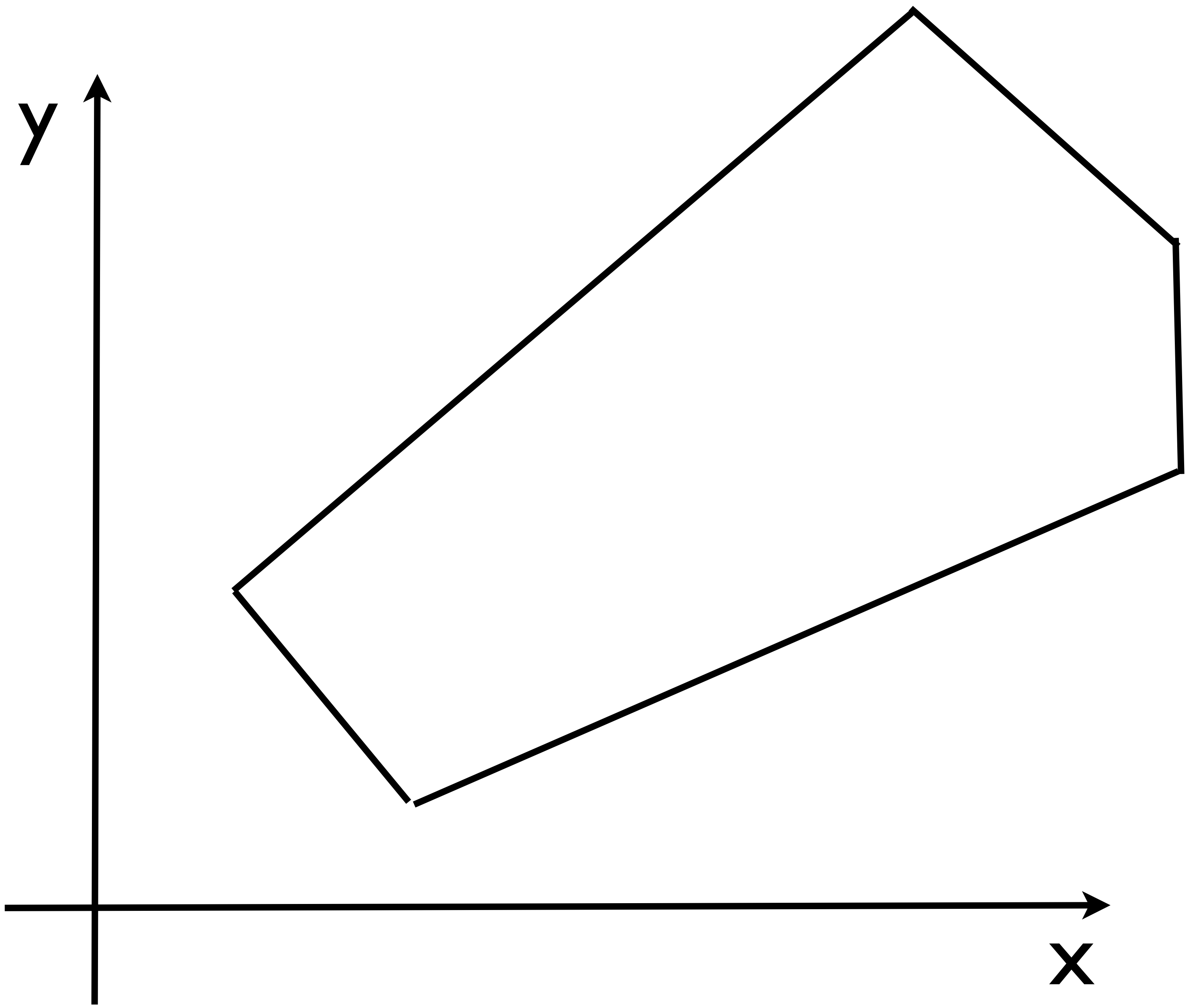
An example of time elapsing

Assume $x^{\bullet}=[1,2]$ and $y^{\bullet}=1$

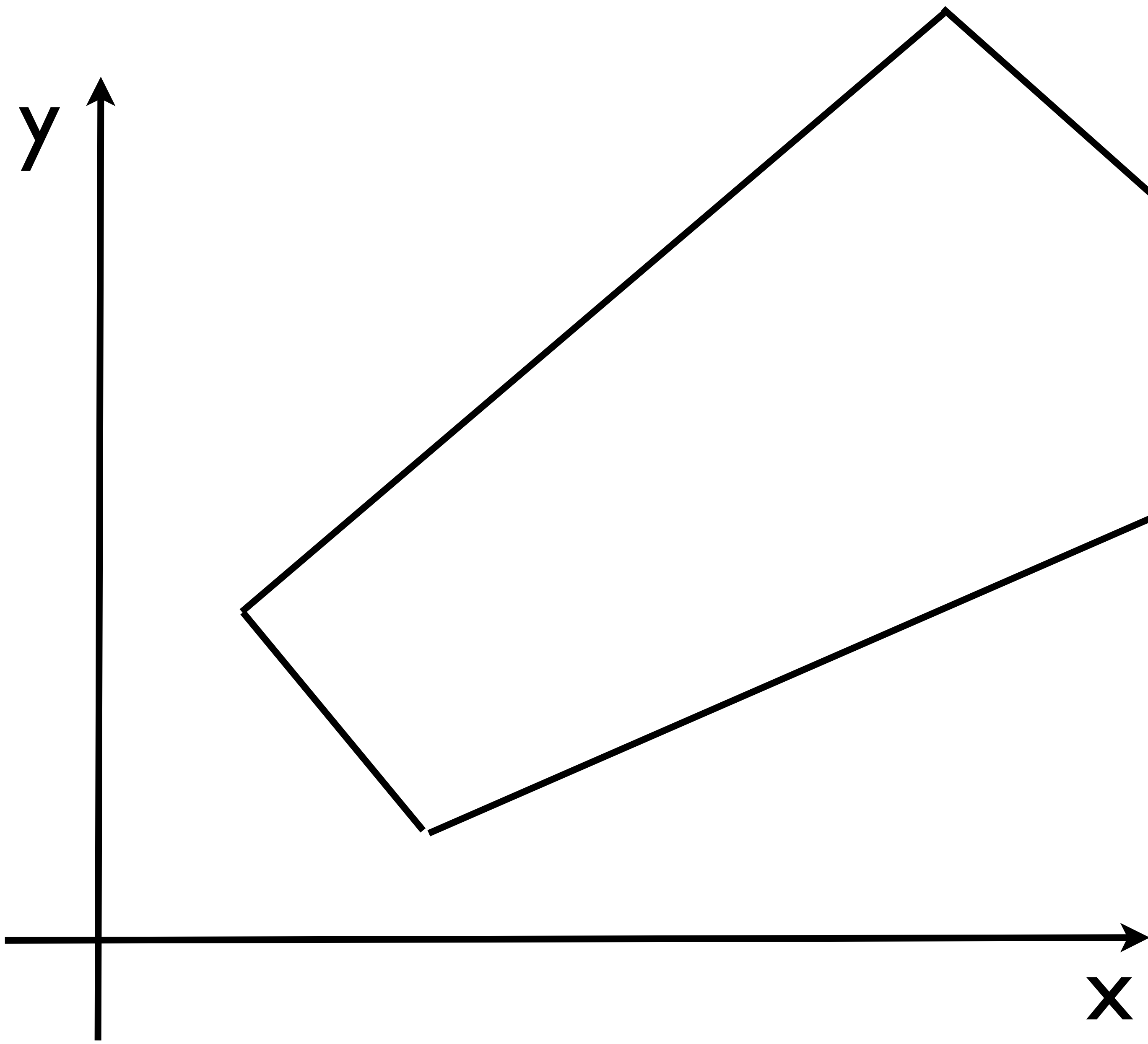


An example of time elapsing

Assume $\dot{x}=[1,2]$ and $\dot{y}=1$

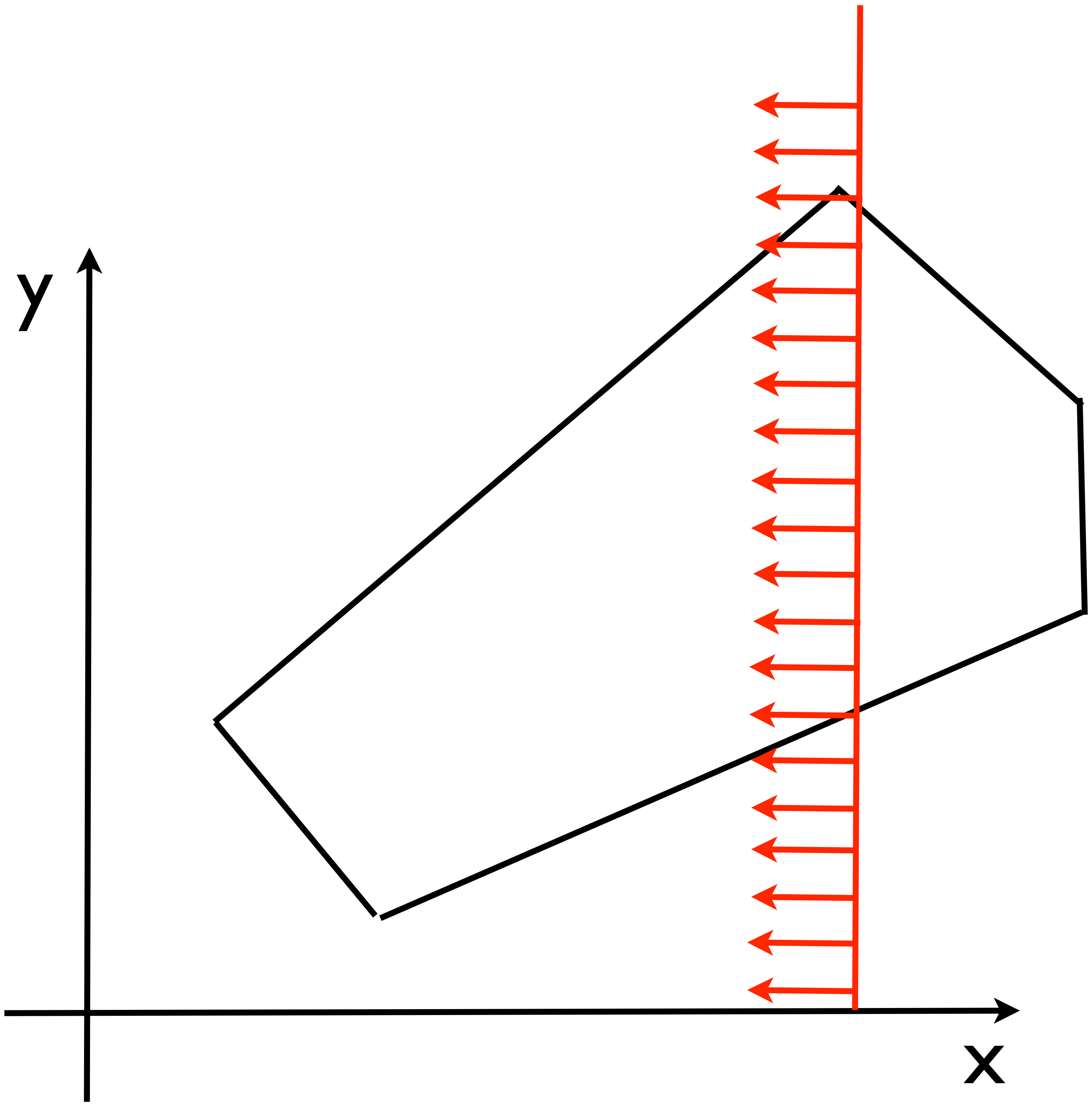


An example of time elapsing



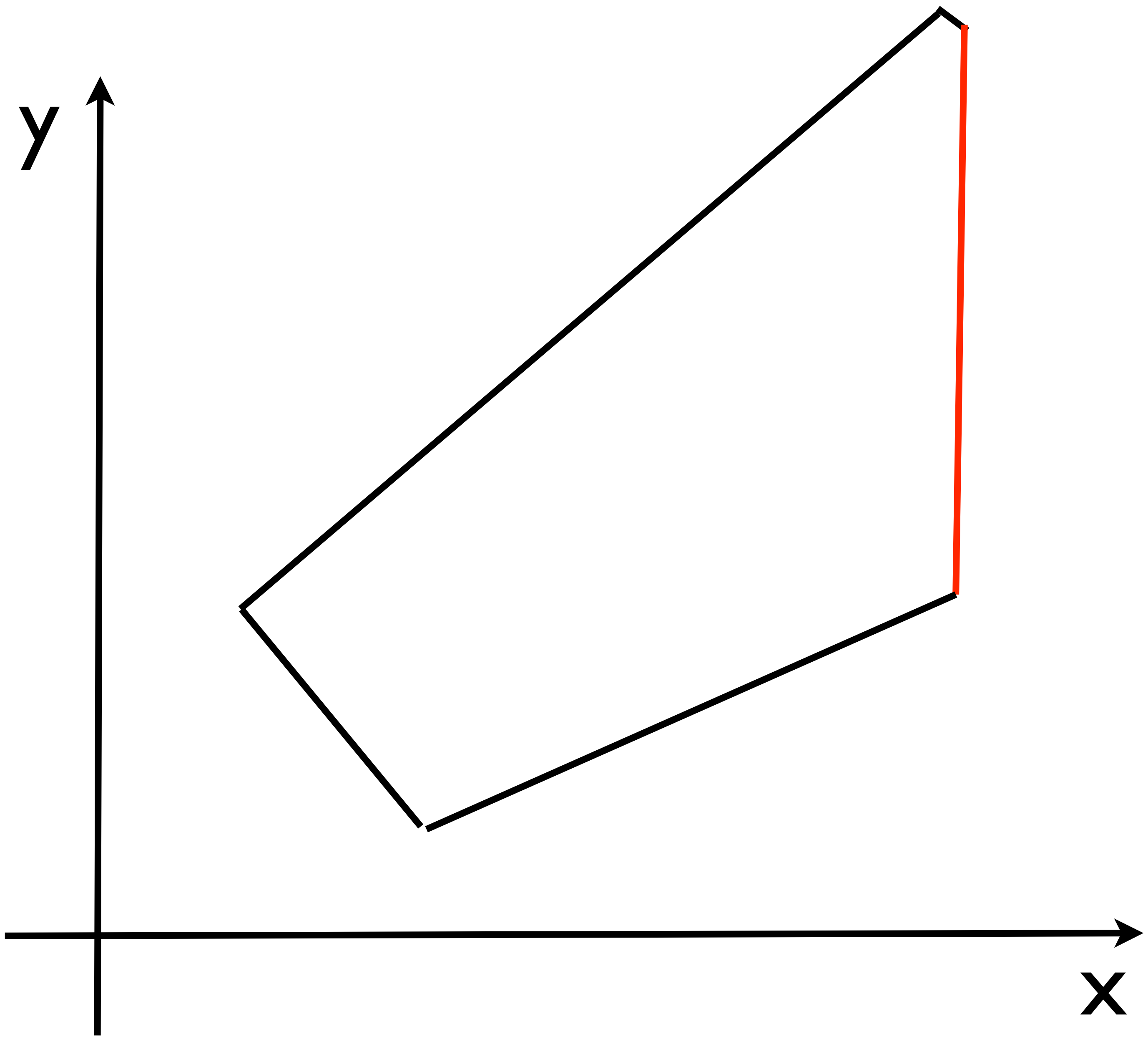
Assume a transition with guard x≤5 and reset of y to zero.

An example of discrete step



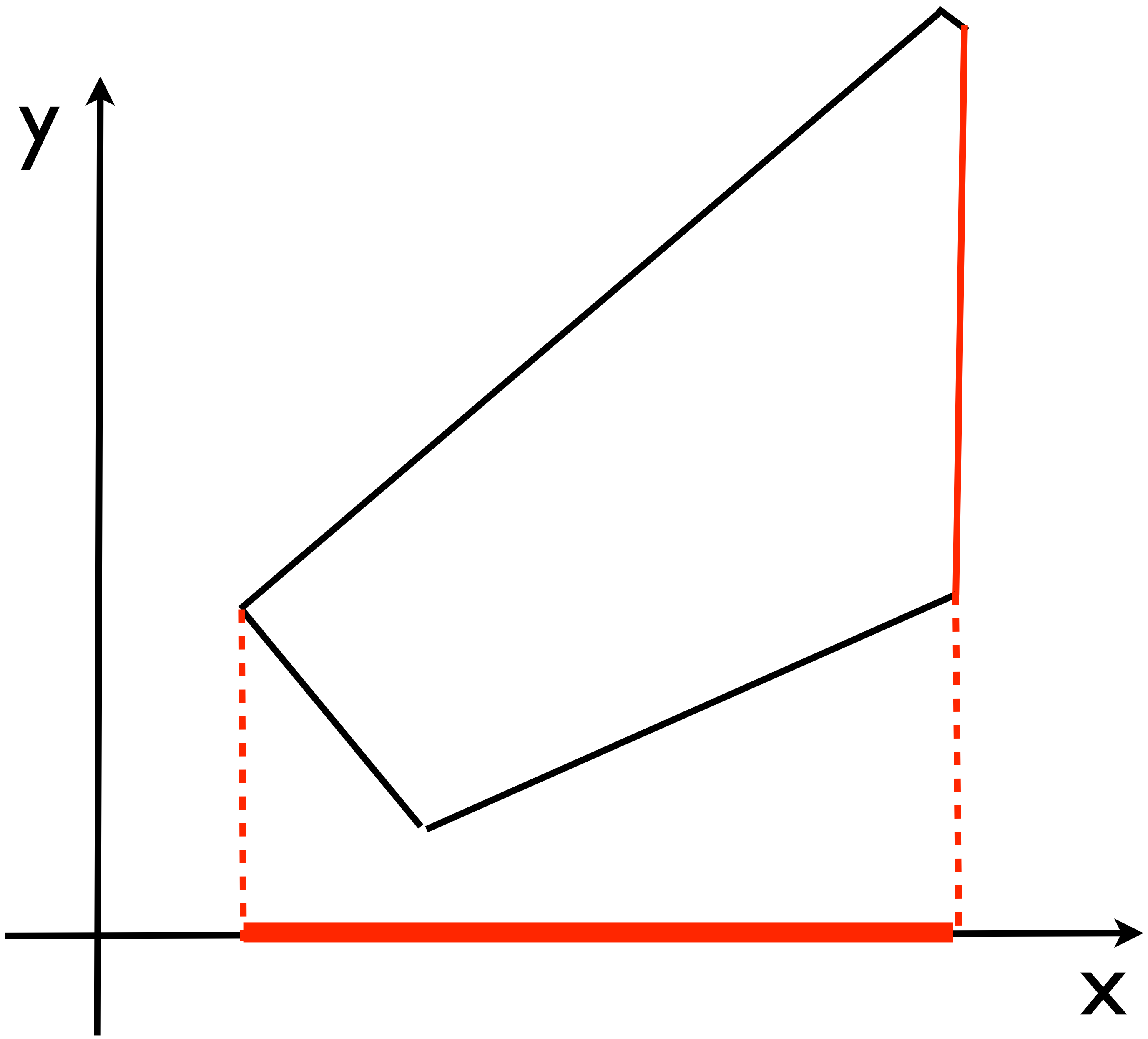
Assume a transition with guard $x \leq 5$ and reset of y to zero.

An example of discrete step



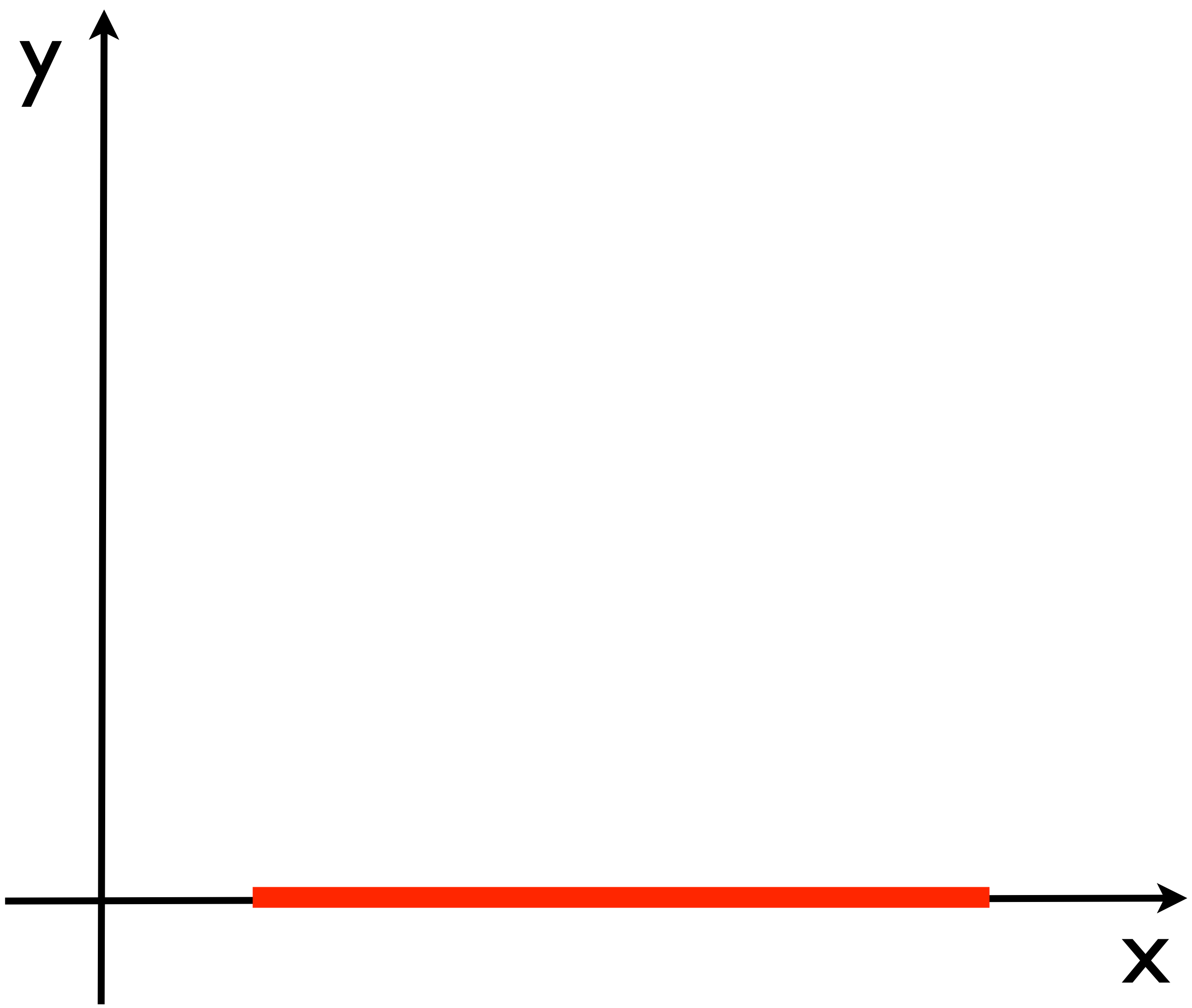
Assume a transition with guard $x \leq 5$ and reset of y to zero.

An example of discrete step



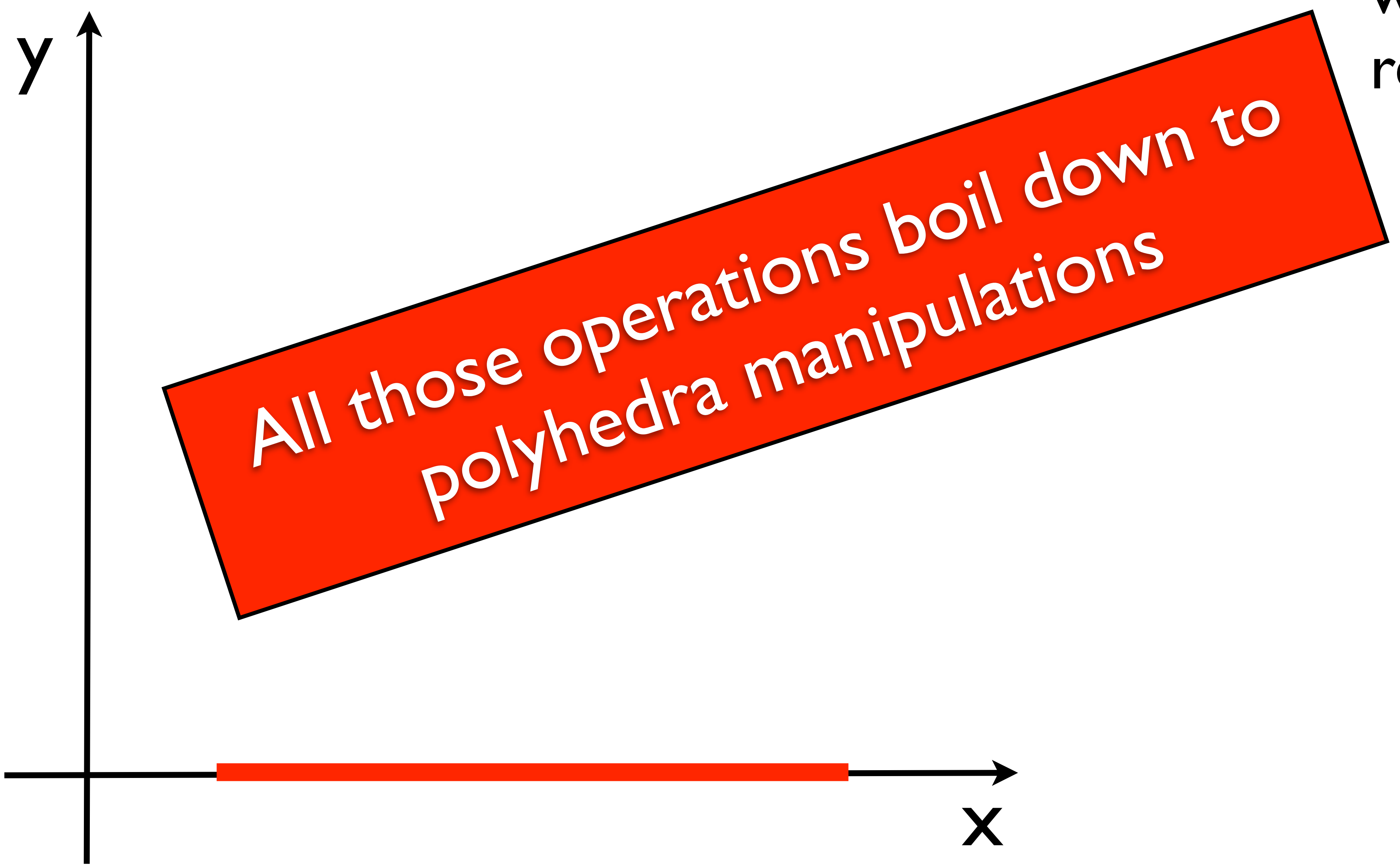
Assume a transition with guard $x \leq 5$ and reset of y to zero.

An example of discrete step



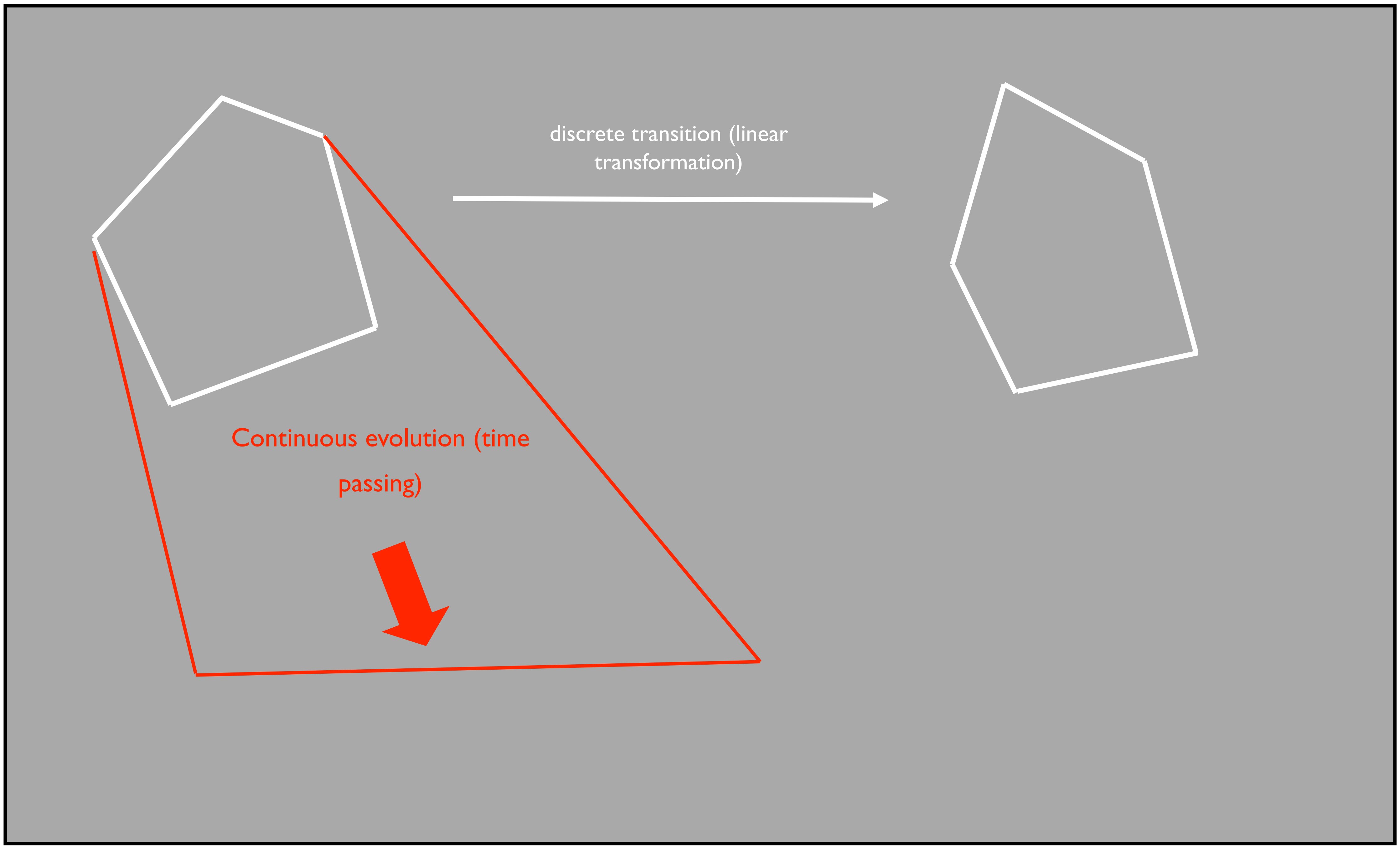
Assume a transition with guard $x \leq 5$ and reset of y to zero.

An example of discrete step

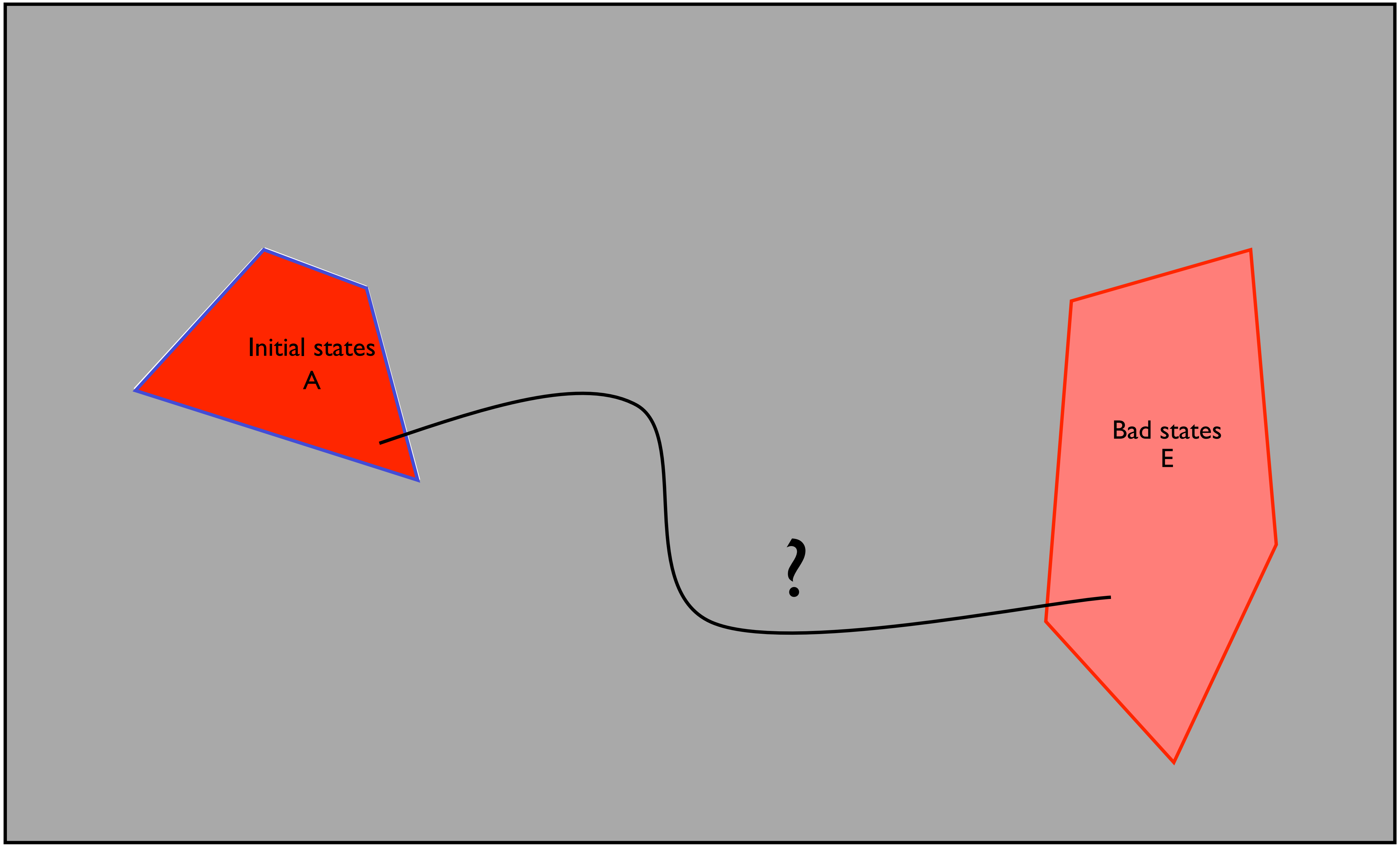


Assume a transition with guard $x \leq 5$ and reset of y to zero.

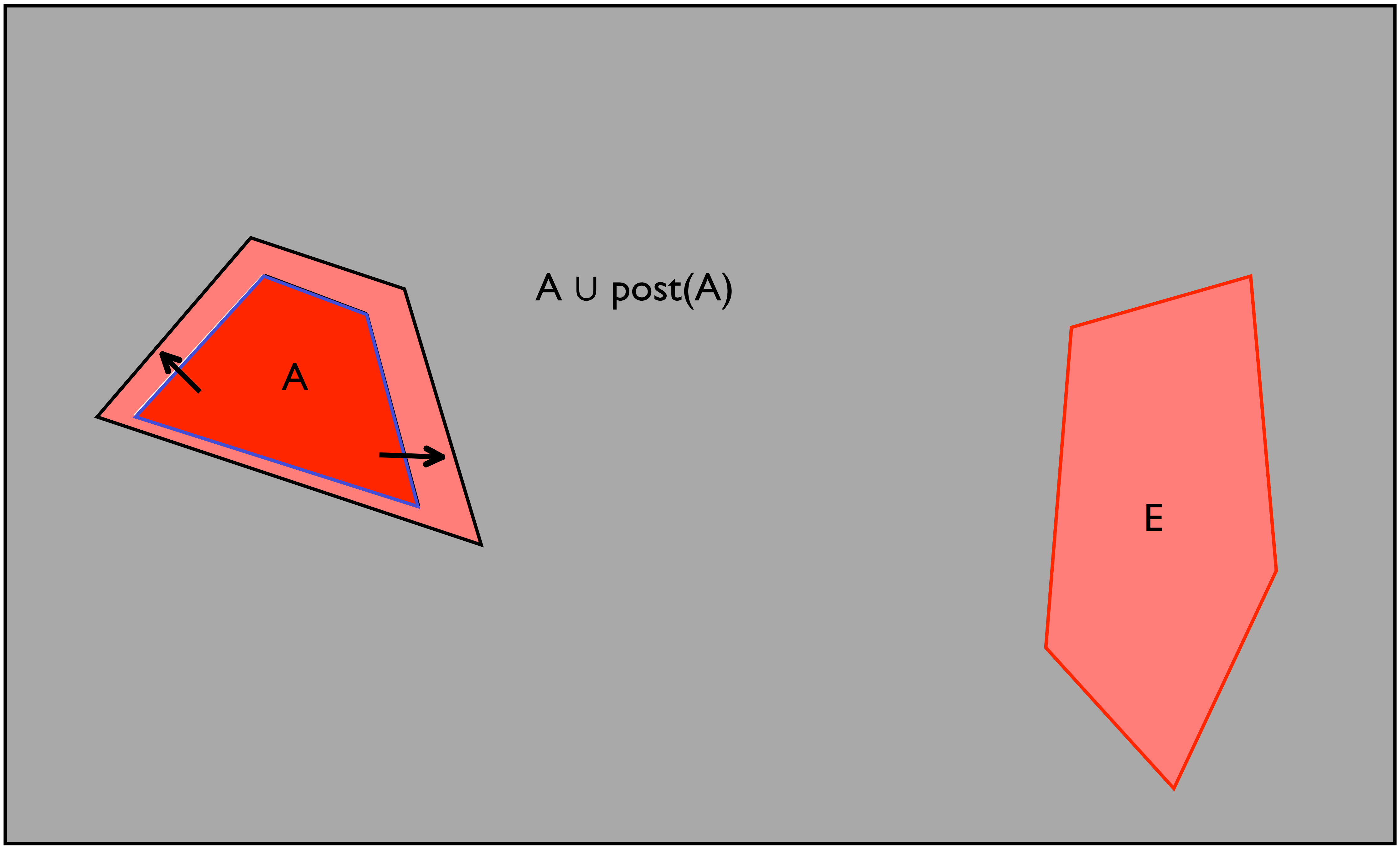
Forward reachability analysis



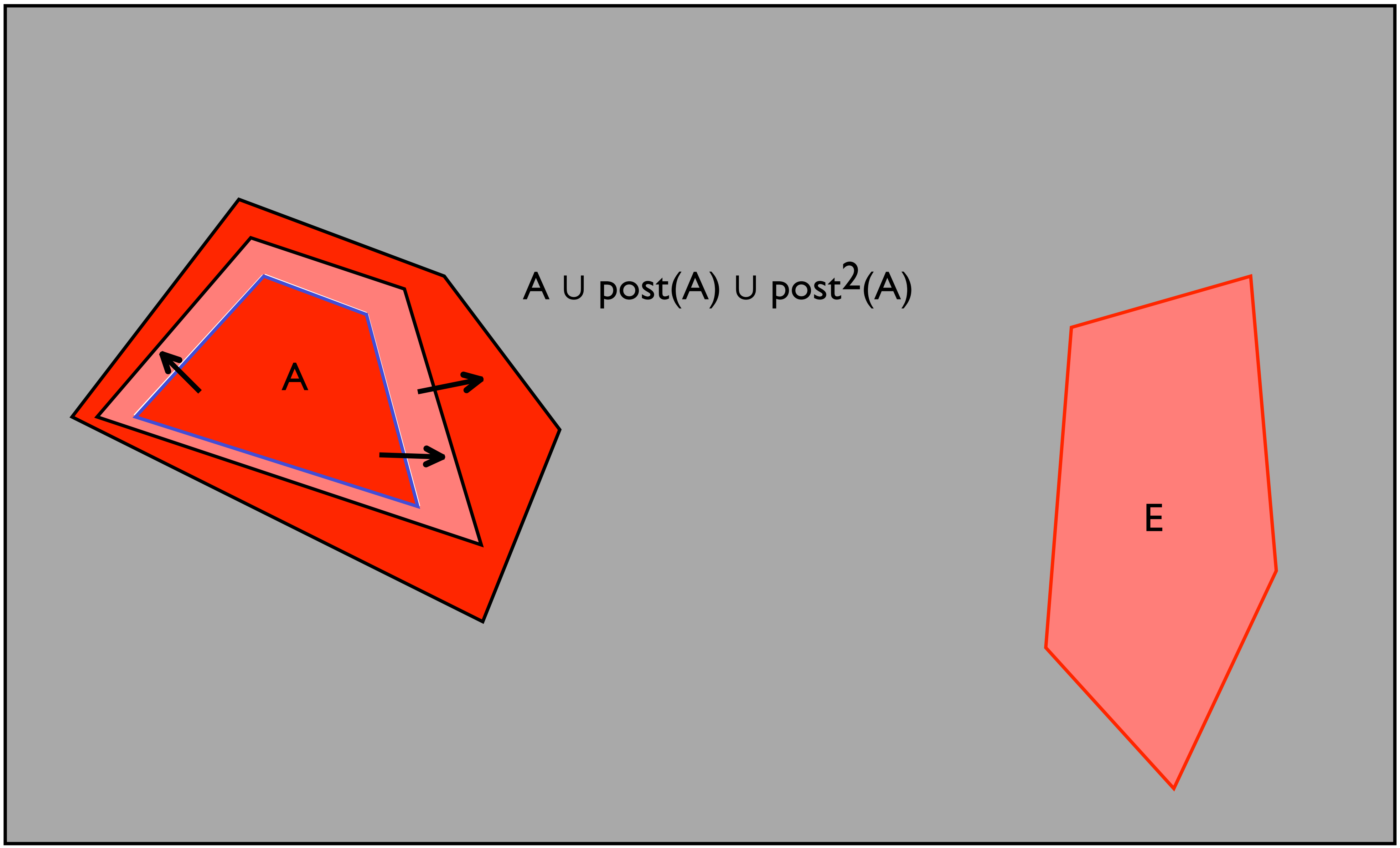
Forward reachability analysis



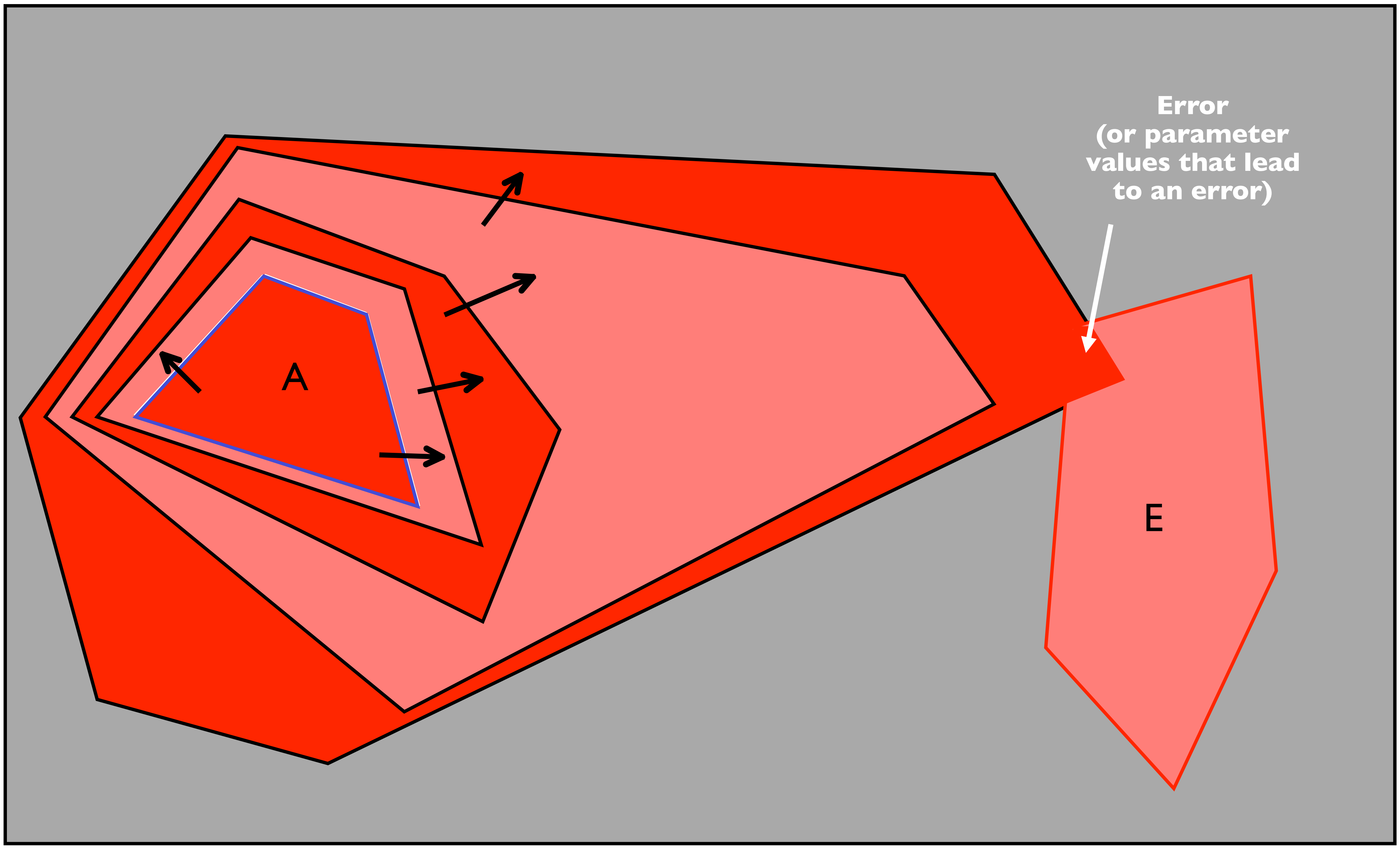
Forward reachability analysis



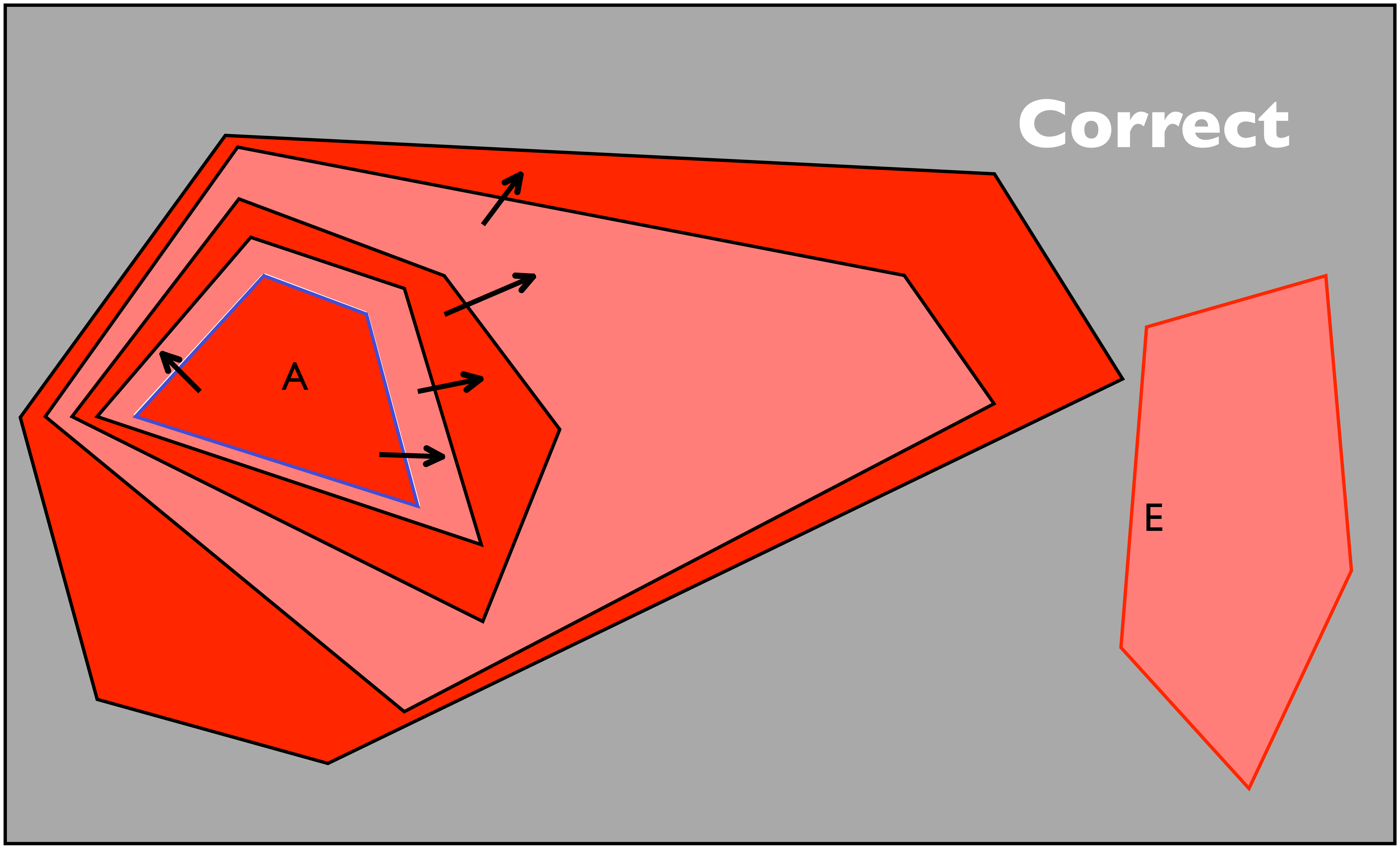
Forward reachability analysis



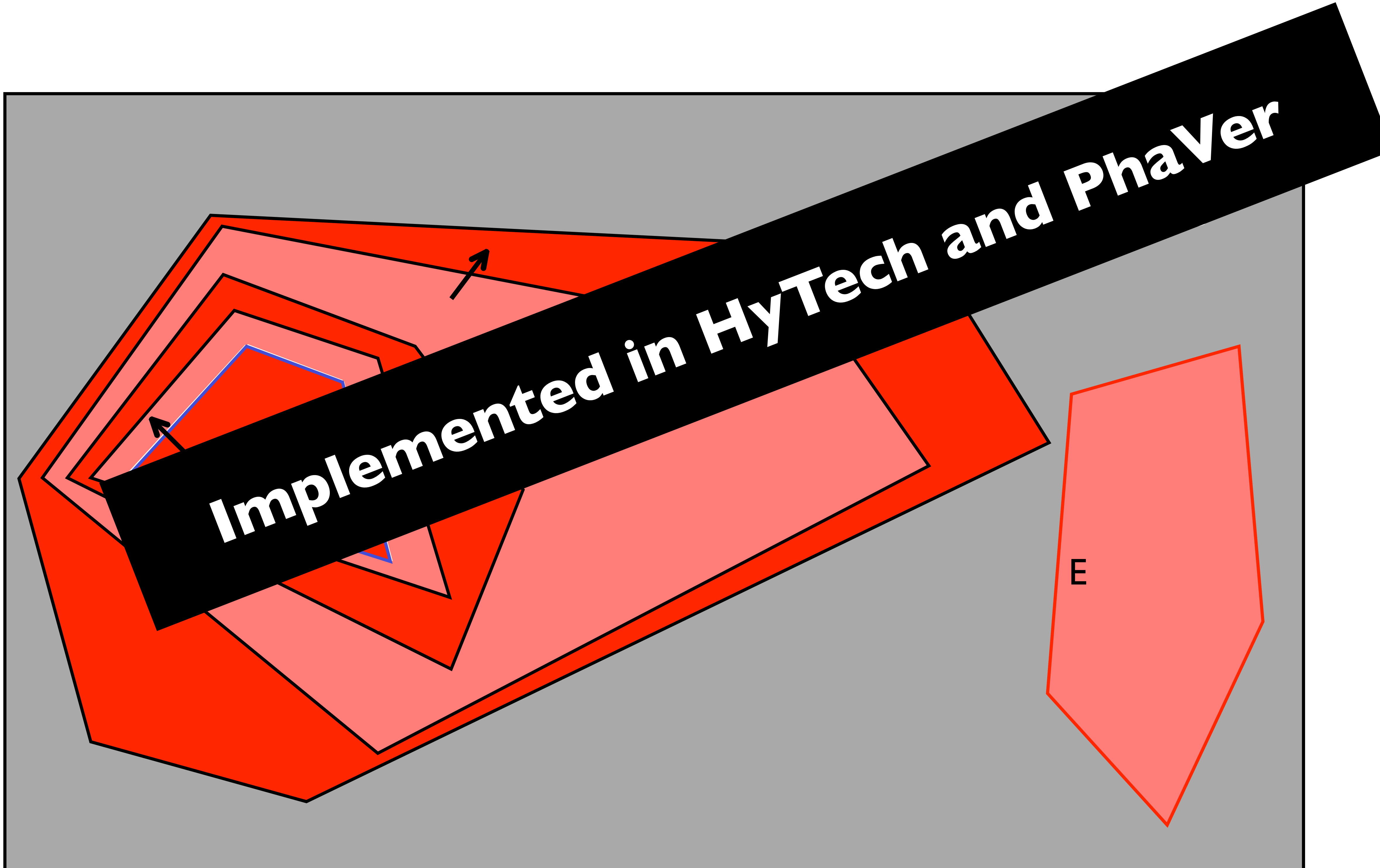
Forward reachability analysis



Forward reachability analysis



Forward reachability analysis



Decidability/ undecidability

Undecidability

Theorem.

The reachability problem for rectangular hybrid automata is undecidable.

This is already the case for **stopwatch automata** ($\dot{x}$=0/1).

Undecidability

Theorem.

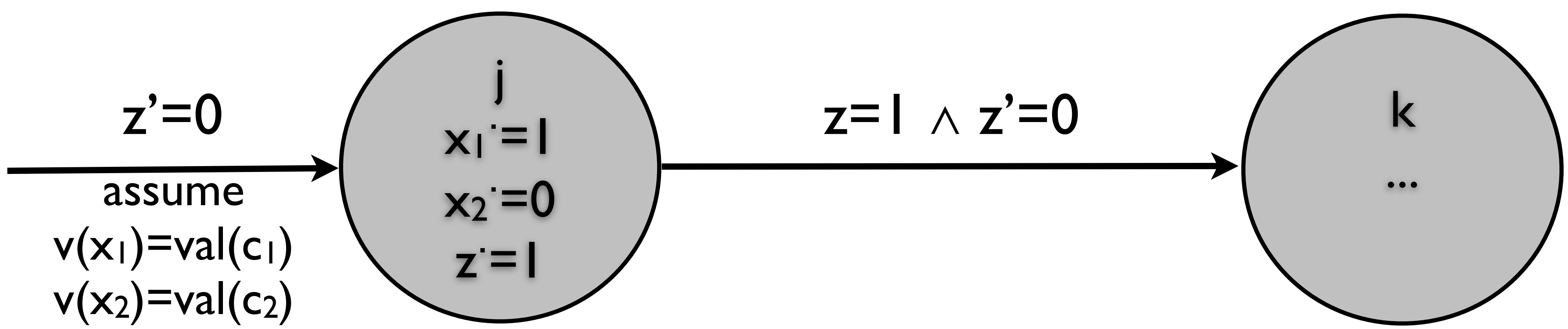
The reachability problem for rectangular hybrid automata is undecidable.

This is already the case for **stopwatch automata** ($\dot{x}$=0/1).

Proof (sketch). By simulation of **two-counter machines** for which the halting problem is undecidable.

To simulate a 2-CM M, we use a RHA with 3 continuous variables.

Let us consider the instruction **j: c_1:=c_1+1; goto k;**

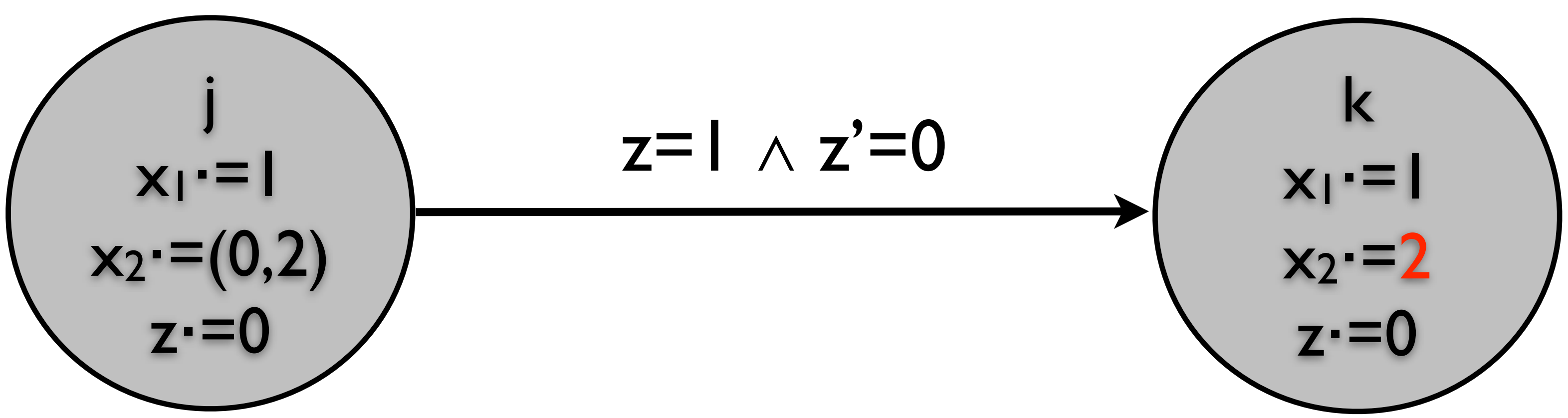


Initialized RHA

- A RHA is **initialized**, if for all discrete jumps (l_1,σ,l_2), and for all variables $x \in X$:
  - either the flow constraints on x in l_1 and l_2 are identical
  - or variable x is updated during the discrete jump from l_1 to l_2

Initialized RHA

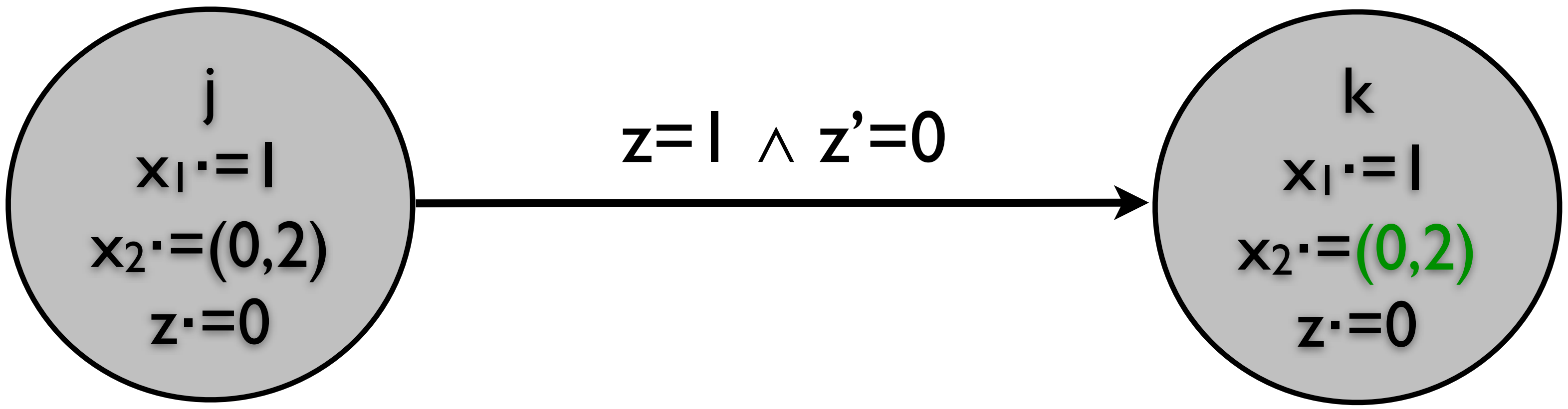
- A RHA is **initialized**, if for all discrete jumps (l_1,σ,l_2), and for all variables $x \in X$:
  - either the flow constraints on x in l_1 and l_2 are identical
  - or variable x is updated during the discrete jump from l_1 to l_2



is not initialized

Initialized RHA

- A RHA is **initialized**, if for all discrete jumps (l_1,σ,l_2), and for all variables x∈X:
  - either the flow constraints on x in l_1 and l_2 are identical
  - or variable x is updated during the discrete jump from l_1 to l_2

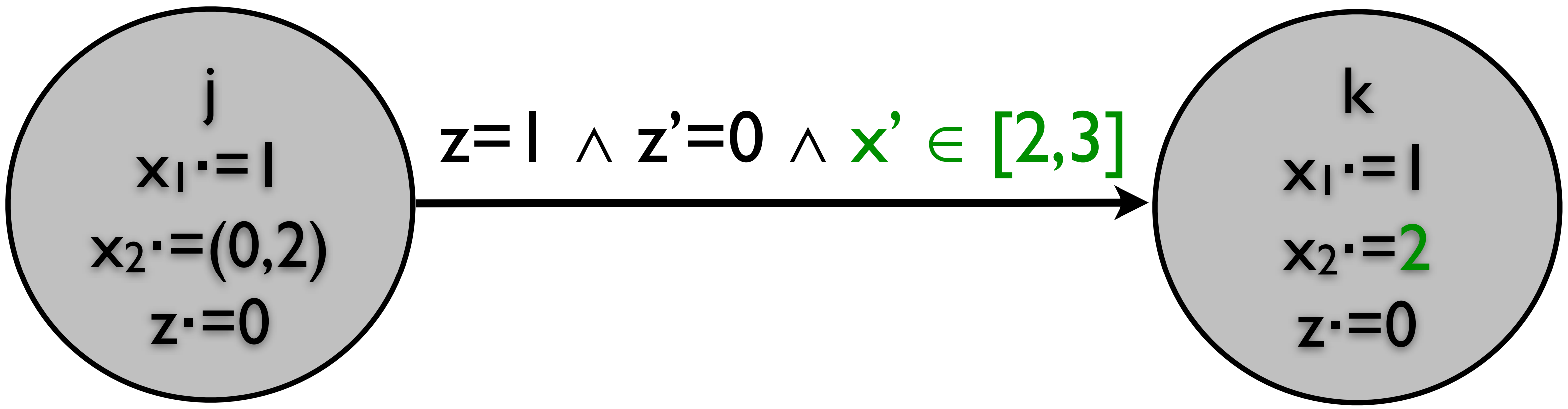


is initialized

Initialized RHA

- A RHA is **initialized**, if for all discrete jumps (l_1,σ,l_2), and for all variables $x \in X$:

  - either the flow constraints on x in l_1 and l_2 are identical
  - or variable x is updated during the discrete jump from l_1 to l_2



is initialized

Initialized RHA

- A RHA is **initialized**, if for all discrete jumps (l_1,σ,l_2), and for all variables $x \in X$:

  - either the flow constraints on x in l_1 and l_2 are identical
  - or variable x is updated during the discrete jump from l_1 to l_2

  Theorem[HPV96]. The reachability problem (and LTL model-checking problem) is **decidable** for the class of **initialized rectangular automata**.

- Note that Initialized RHA generalizes timed automata

- Existence of finite similarity quotient (init-RHA) and bisimilarity quotient (TA)

Decidability/Undecidability

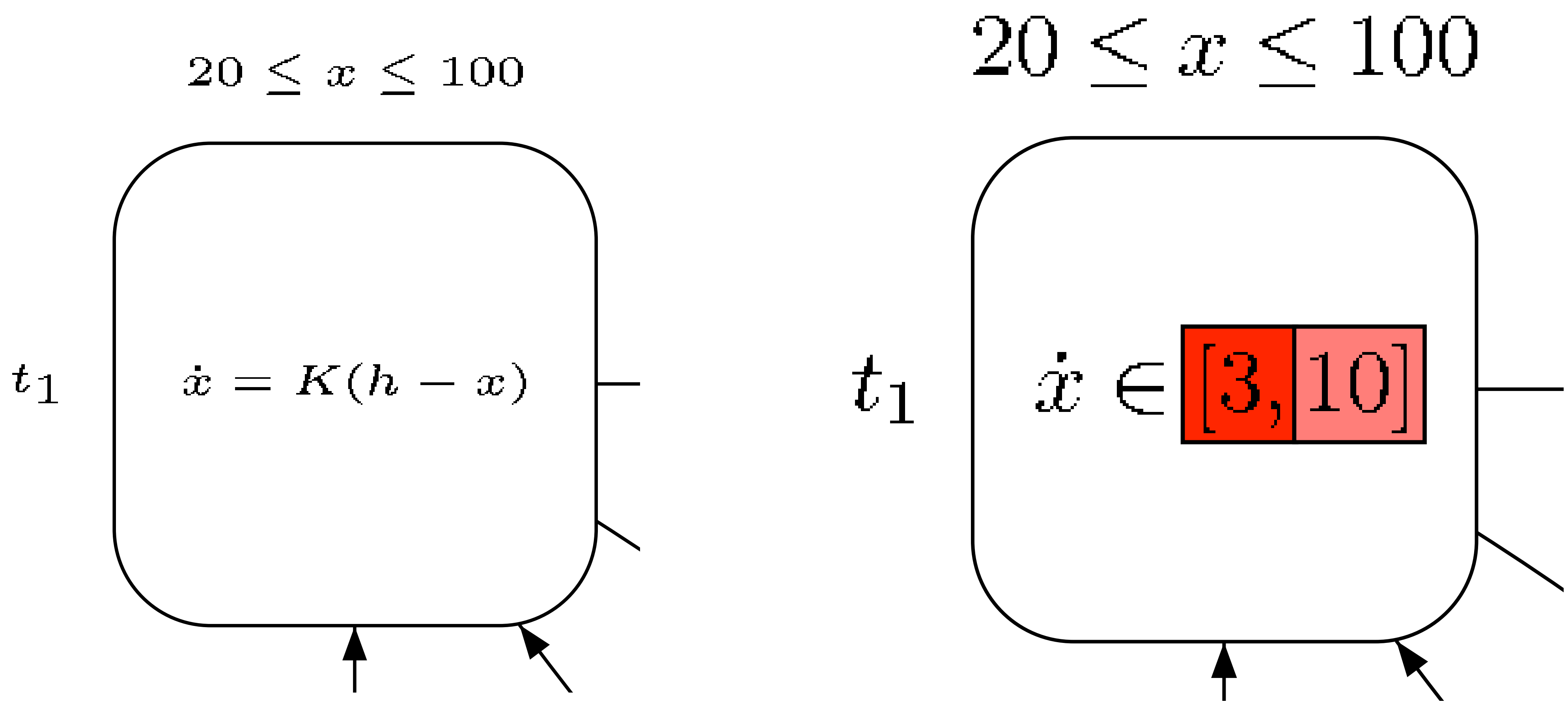
	Reach
Timed automata	☝
Initialized RHA	☝
RHA	☟ **(Stopwatch)**
LHA	☟
Affine HA	☟
O-Minimal HA	☝

Beyong RHA/LHA Approximate Reachability

Rectangular approximations

- **Approximate** complex dynamics with rectangular dynamics
- ... use PhaVer or Hytech for analysis
- Rectangular approximations are often **precise enough**
- For each control mode we **partition** the space into rectangular regions
- Within each region, the flow field is **over-approximated** using rectangular flows
- Those approximations can often be obtained automatically:
  for affine HA $\rightarrow$ solve an **LP** problem
- Approximations can be made **arbitrarily precise** by approximating over suitably small regions of the state space

An example



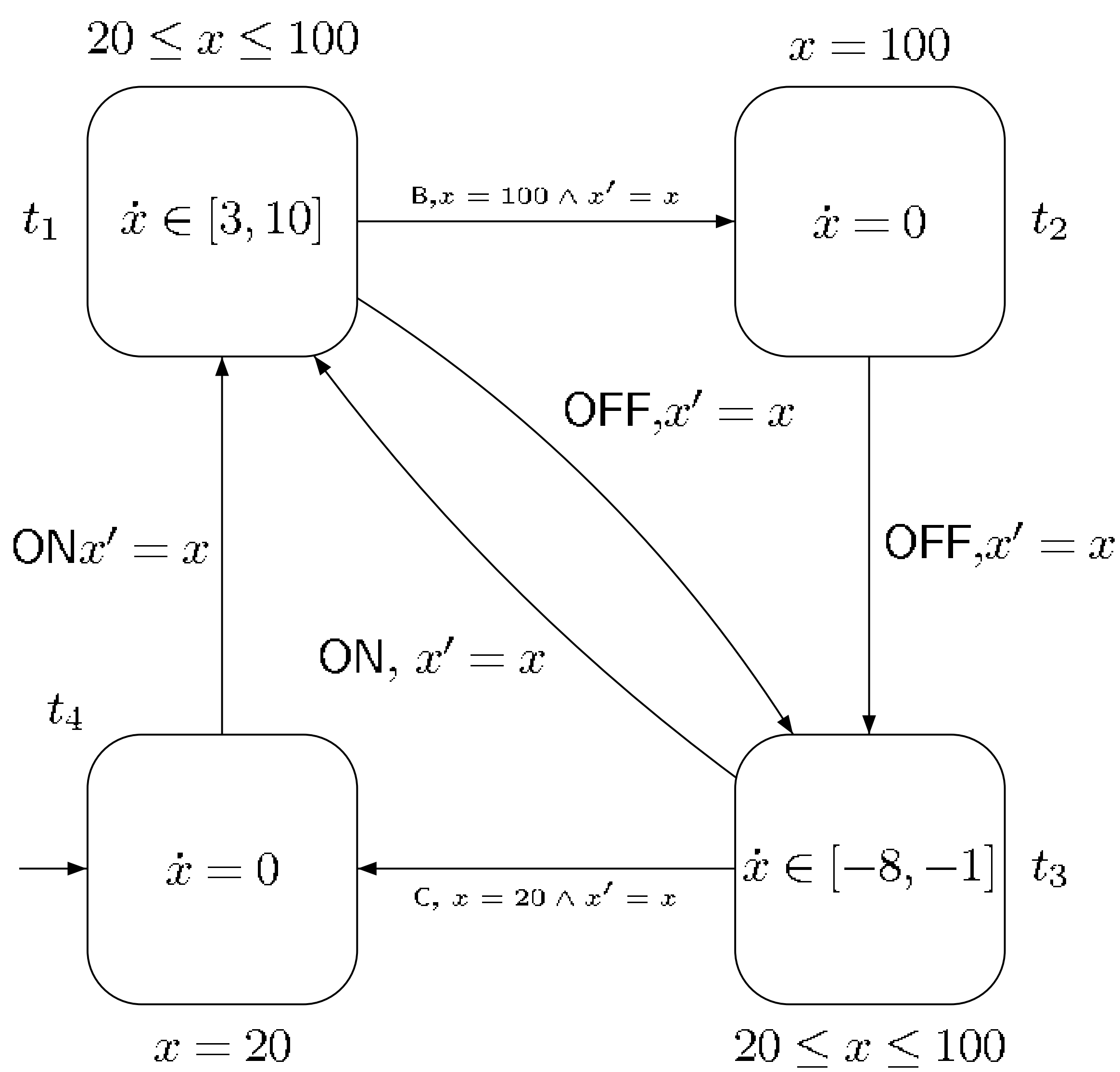
$\text{Max}_{x \in [20,100]}$ K(h-x) = K(h-20) = 0.075(150-20) = 9.75 ≤ 10

$\text{Min}_{x \in [20,100]}$ K(h-x) = K(h-100) = 0.075(150-100) = 3.75 ≥ 3

These are LPs

An example

- Applying this computation for each location, we get the following **rectangular approximation** of the tank:



Over-approximations and correctness

- Let us note **RectOver**(H) the rectangular over-approximation obtained using the previous method;
- RectOver(H) is a **over-approximaiton** of the original system in the following sense:

$$\mathbf{Path_F}(\llbracket \mathbf{H} \rrbracket) \subseteq \mathbf{Path_F}(\llbracket \mathbf{RectOver(H)} \rrbracket)$$

- **Transfert of correctness** from overapproximations:

if $\mathbf{Path_F}(\llbracket \mathbf{RectOver(H)} \rrbracket) \cap \mathbf{BadPaths} = \emptyset$
then $\mathbf{Path_F}(\llbracket \mathbf{H} \rrbracket) \cap \mathbf{BadPaths} = \emptyset$

Over-approximations and correctness

- When over-approximating the behavior of a system, we face the possibility to get **false negatives** during verification;
- Indeed, the set of behaviors of the **over-approximation** is a **superset** of the behaviors of the original system...
- ...so if we have that

$$\mathbf{Path_F([\![RectOver(H)]\!]) \cap BadPaths \neq \emptyset}$$

it is **not** nessarily the case that

$$\mathbf{Path_F([\![H]\!]) \cap BadPaths \neq \emptyset}$$

Candidate counter examples

- A path $\lambda = s_0 \tau_0 s_1 \tau_1 \ldots \tau_{n-1} s_n$ is an **candidate counter example** if
  - $\lambda \in [\![\text{OverRect}(H)]\!] \cap \text{BadPaths}$
- When facing a candidate counter example, we check if the counter example is realizable in the original model, so we ask:
  - $\lambda \in^{?} [\![H]\!]$

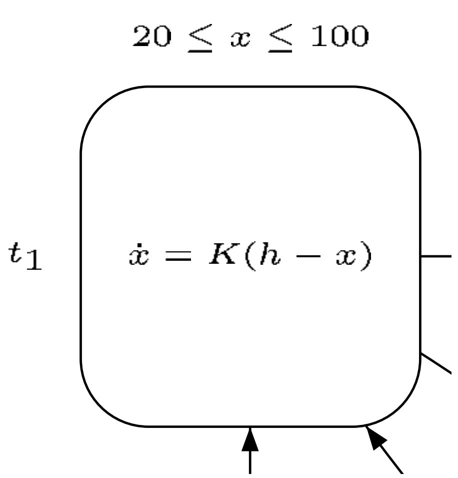
    This test is possible for larger class than rectangular automata, i.e. affine/polynomial hybrid automata.
- If $\lambda \in [\![H]\!]$, then we have found a **real** counter example i.e., the a Bad path in the original HA H.

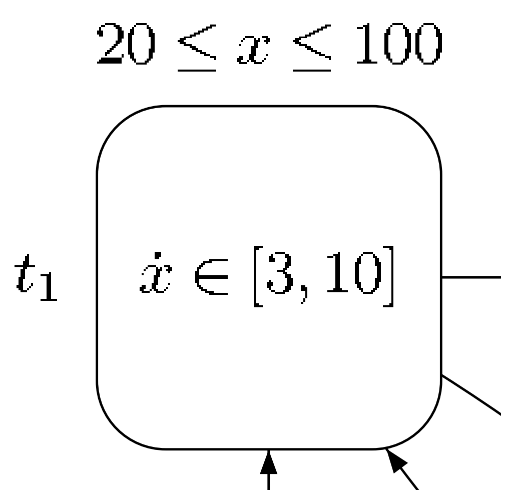
Spurious counter-examples

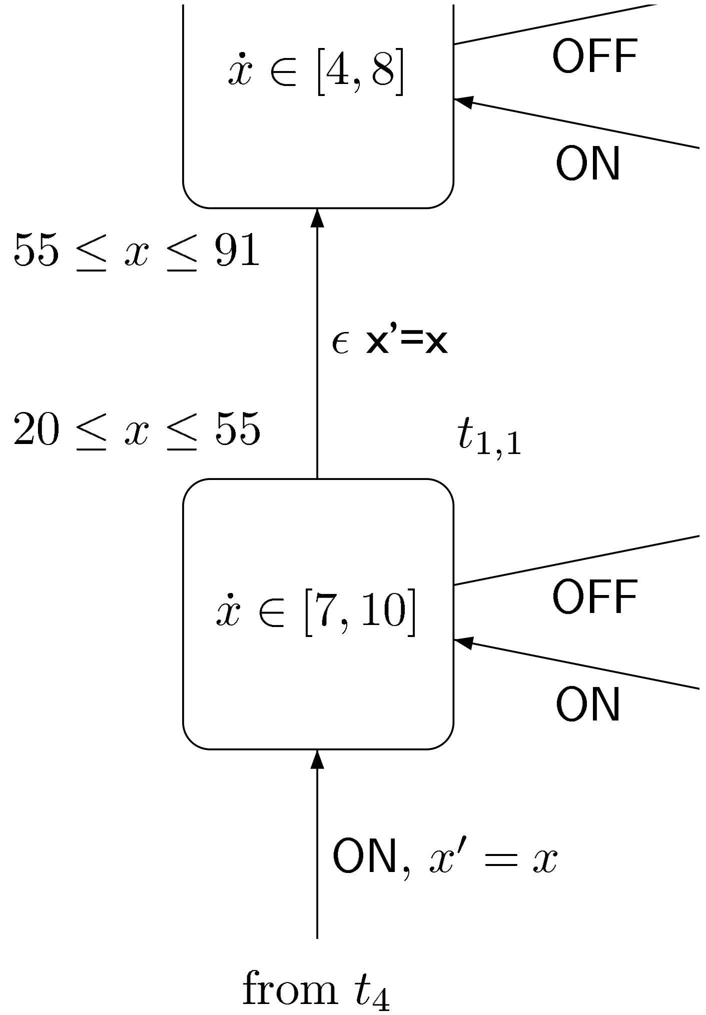
- If $\lambda \notin [\![H]\!]$, then λ is a **spurious counter example** i.e.:
  - $\lambda \in [\![\text{OverRect}(H)]\!] \cap \text{BadPaths}$
  - $\lambda \notin [\![H]\!]$
- In this case, we must **refine** OverRect(H) in order to eliminate the counter example.
- There is a large research effort in the CAV community on the so-called **counter-example based abstraction refinement**, and variants.

Abstraction refinement

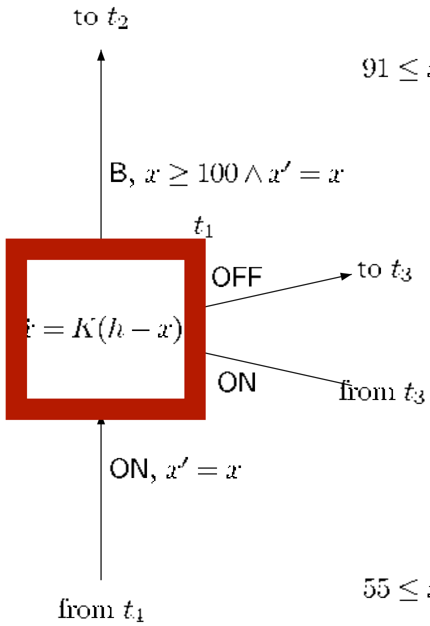
- In presence of **spurious counter examples**, we **refine** the rectangular approximation by **splitting** locations to decorate them with smaller rectangular regions.



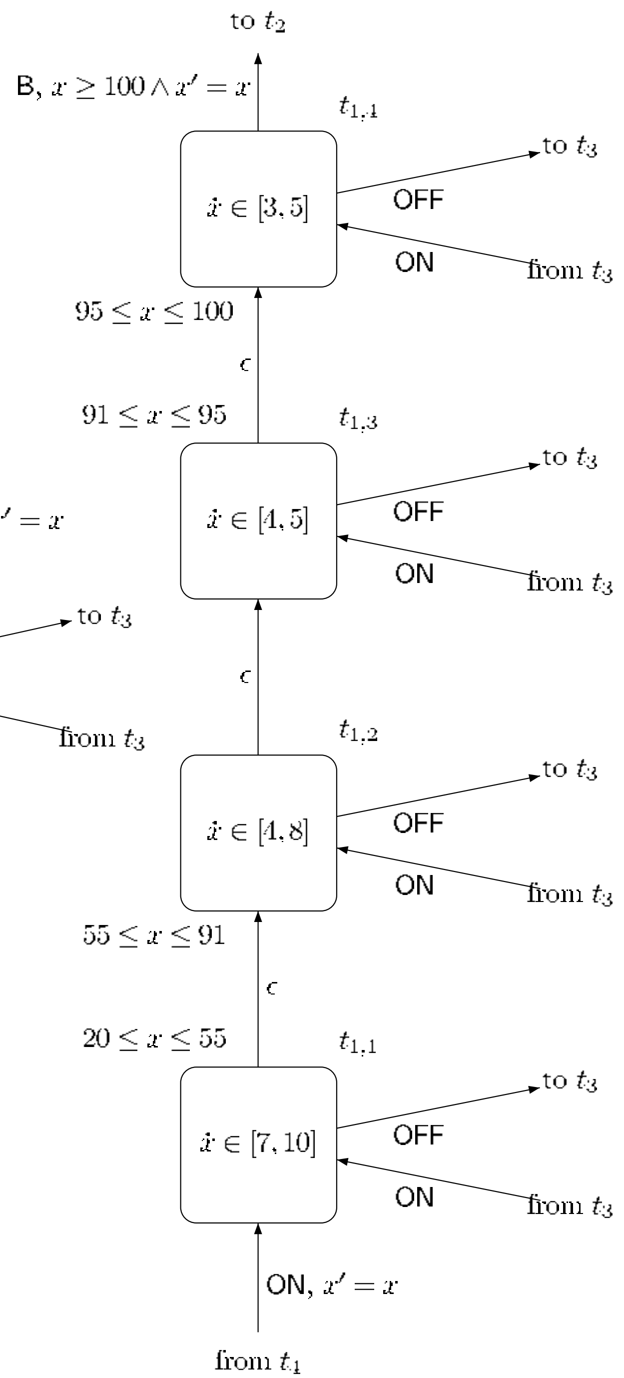




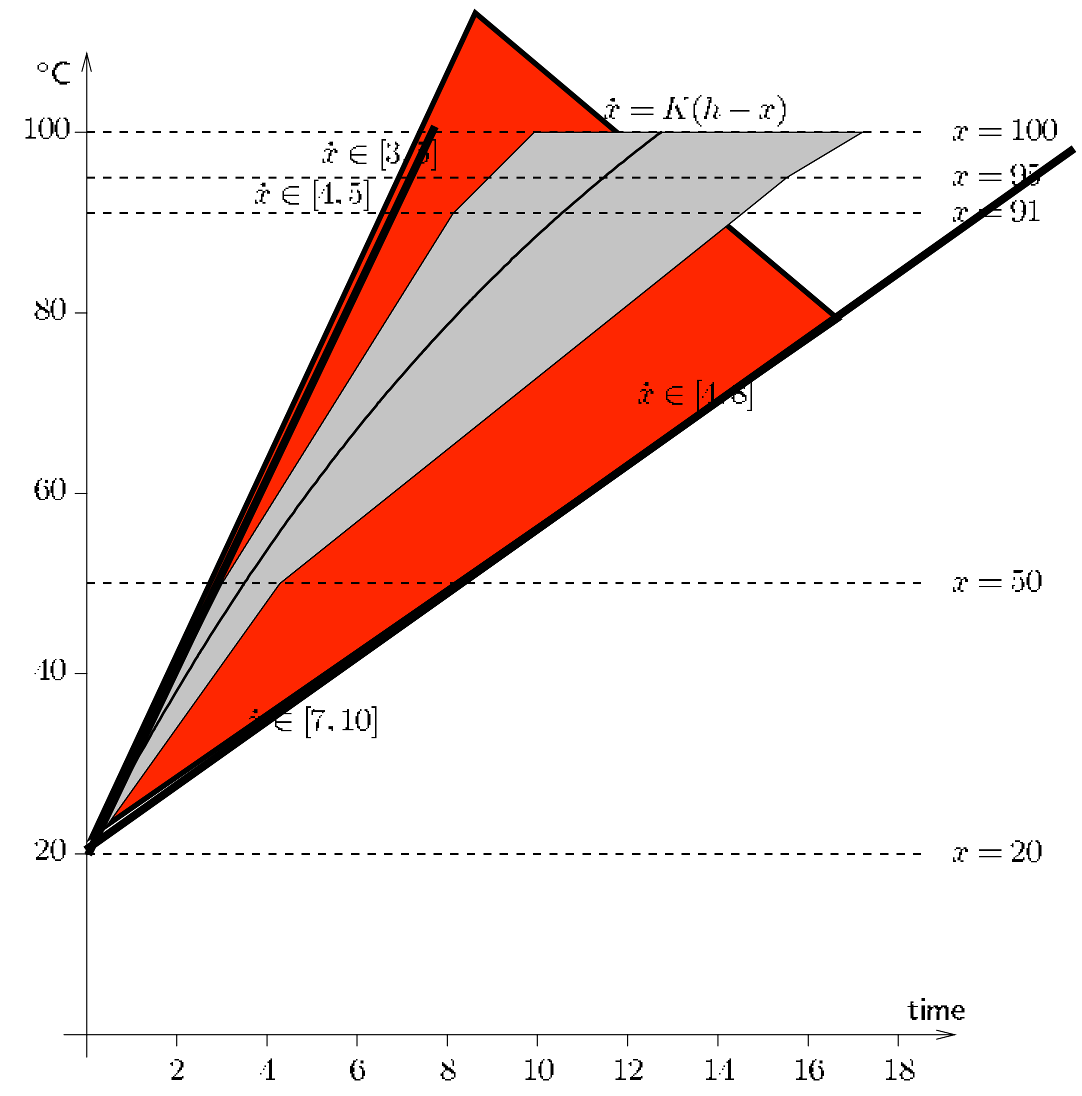
Example



to t_2
B, $x \geq 100 \wedge x' = x$
t_1
OFF
to t_3
$\dot{x} = K(h-x)$
ON
from t_3
ON, $x' = x$
from t_1



to t_2
B, $x \geq 100 \wedge x' = x$
$t_{1,4}$
$\dot{x} \in [3,5]$
to t_3
OFF
ON
from t_3
$95 \leq x \leq 100$
c
$91 \leq x \leq 95$
$t_{1,3}$
$\dot{x} \in [4,5]$
$t_{1,2}$
$\dot{x} \in [4,8]$
$55 \leq x \leq 91$
$20 \leq x \leq 55$
$t_{1,1}$
$\dot{x} \in [7,10]$
ON, $x' = x$
from t_1



°C
100
80
60
40
20
$\dot{x} = K(h-x)$
$\dot{x} \in [3,5]$
$\dot{x} \in [4,5]$
$\dot{x} \in [4,8]$
$\dot{x} \in [7,10]$
$x = 100$
$x = 95$
$x = 91$
$x = 50$
$x = 20$
time
2
4
6
8
10
12
14
16
18

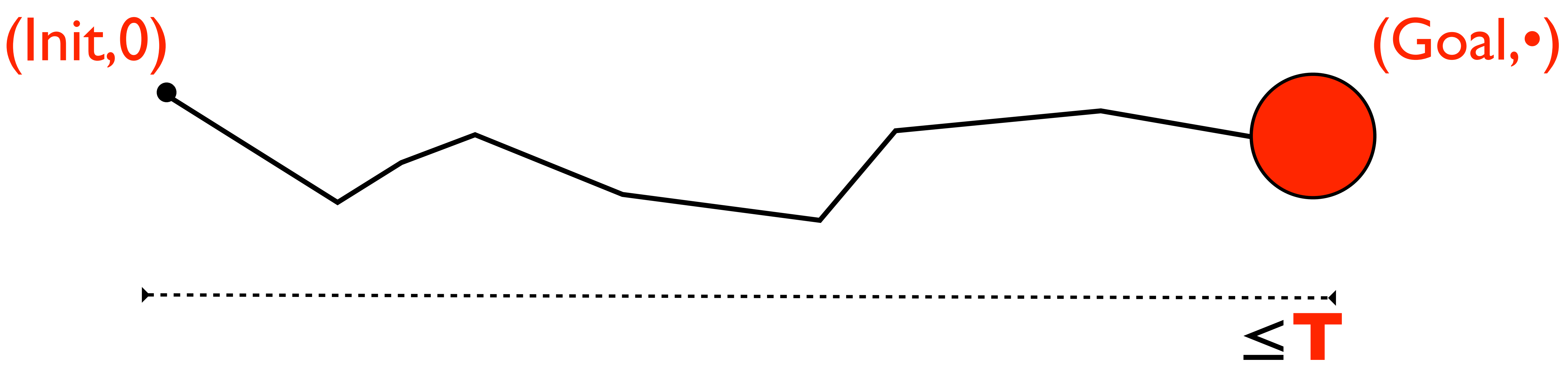
Time-bounded Reachability

Time Bounded Reachability

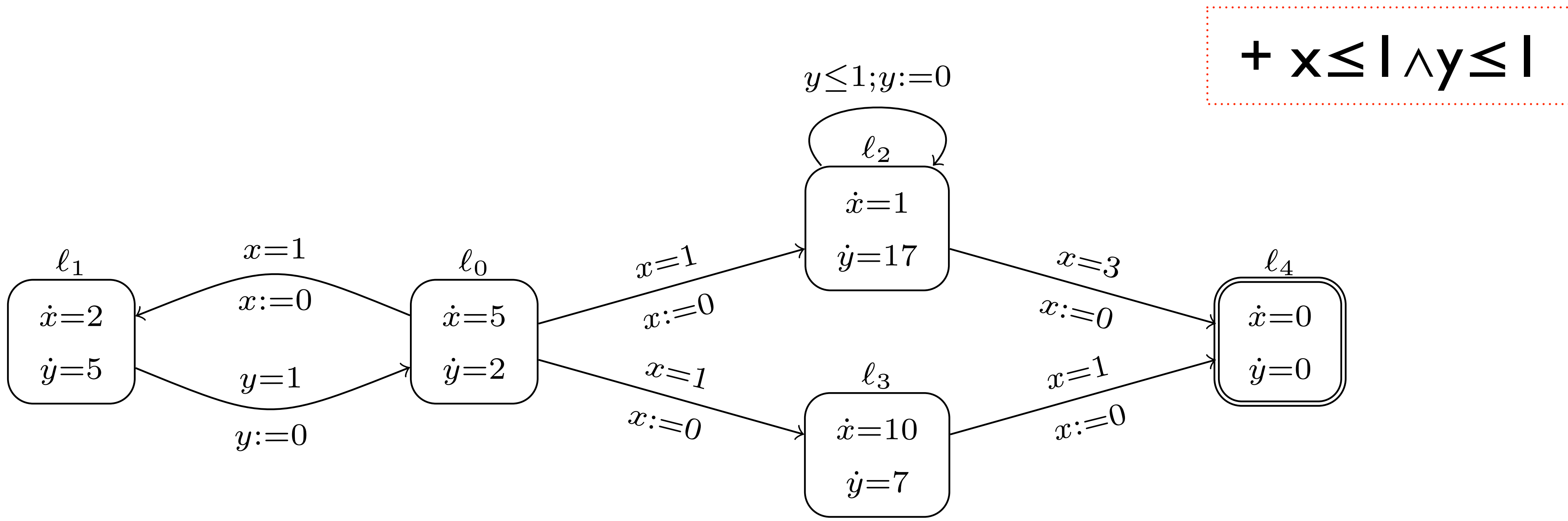
Definition

- Given an LHA H=(X,Loc,Edges,Rates,Inv,Init)
  - a location Goal∈Loc and
  - a time bound **T**∈ℕ

The **time bounded reachability problem** is to decide if ∃ρ=(Init,0)→(Goal,•) of H with duration(ρ)≤**T**.



Time Bounded Reachability



- This automaton is **non-initialized**, but

  (I) **non-negative** rates

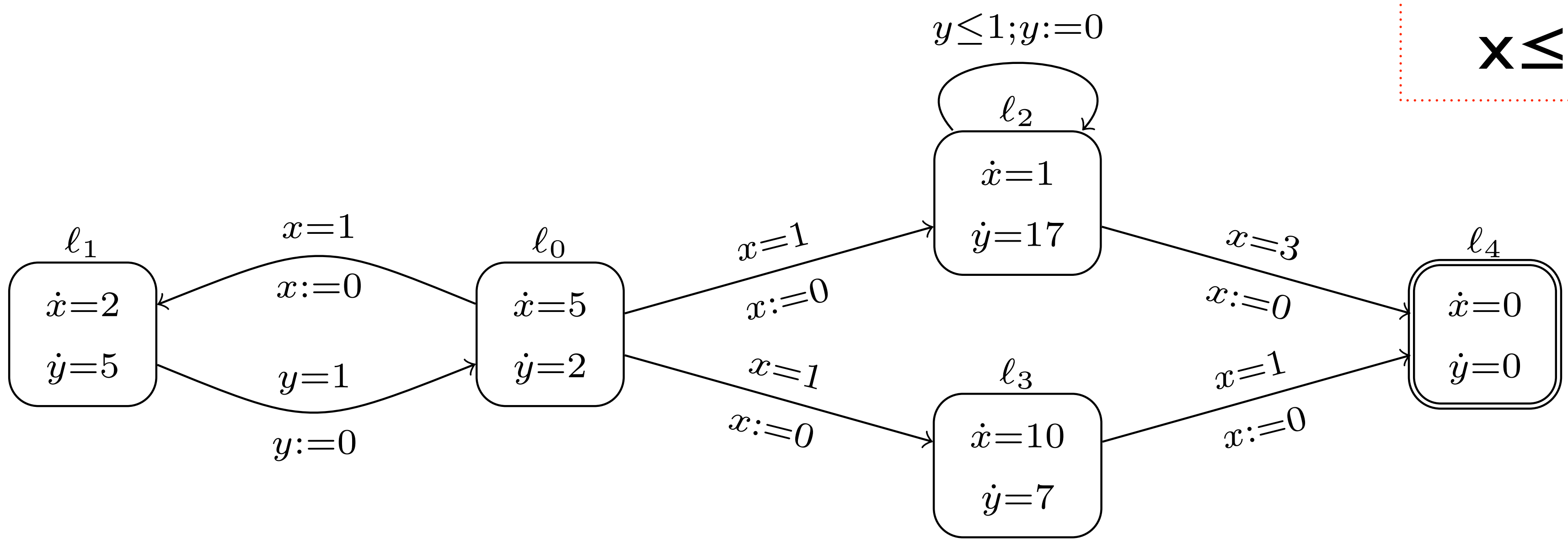
  (II) **diagonal free**

- class **RHA**⊕ for which we show **decidability** of TBR

Time Bounded Reachability

+ Invariant

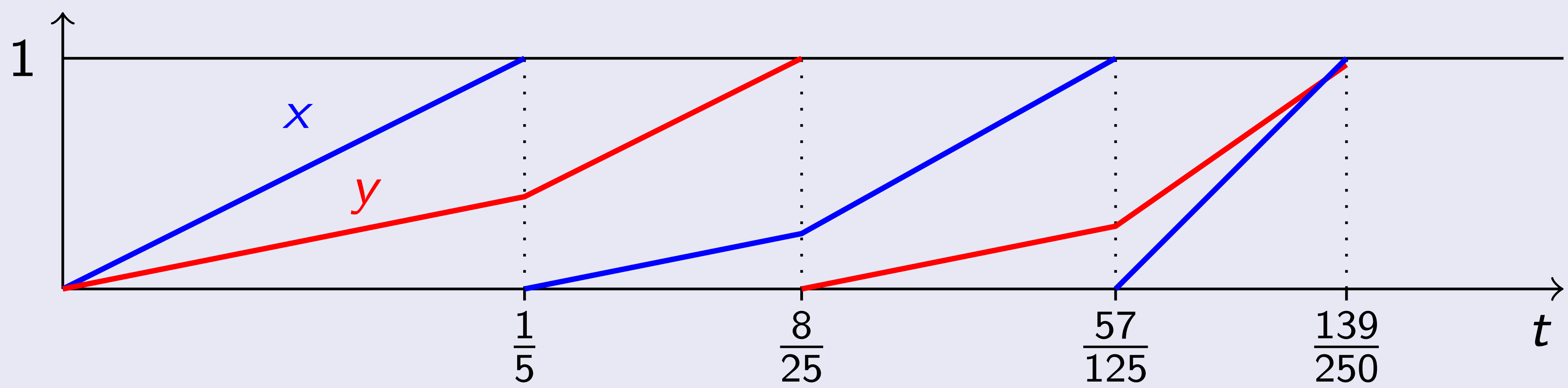
$x \leq 1 \wedge y \leq 1$



Goal

$T \leq 1$

$$(\ell_0,0,0) \xrightarrow{\frac{1}{5}, e_{01}} \left(\ell_1,0,\tfrac{2}{5}\right) \xrightarrow{\frac{3}{25}, e_{10}} \left(\ell_0,\tfrac{6}{25},0\right) \xrightarrow{\frac{17}{125}, e_{03}} \left(\ell_3,0,\tfrac{34}{125}\right) \xrightarrow{\frac{1}{10}, e_{34}} \left(\ell_4,0,\tfrac{243}{250}\right).$$



Additional hypothesis (wlog)

- RHA⊕:
  - **non-negative** rates
  - **diagonal free**

- All variables are bounded by 1
  - (L,2.1,4.7) is encoded by ((L,2,4),0.1,0.7)
  - **Only** guards of the form **x<1**, **x=1**
  - As soon as a clock reaches value 1, it is reset

Bounding the number of transitions

Our goal:

- Given ρ an execution of H reaching Goal from (L_0,x_0) within **T** time units.
- We want to build an execution ρ' of H such that :

  -ρ' reaches Goal from (L_0,x_0) within **T** time units
  -the number of transitions of ρ' is **bounded** by a constant depending only of H and **T**

Solution:

① Simple observation: bounding the number of equalities

② Bounded witness between equalities

Bounding number of equalities

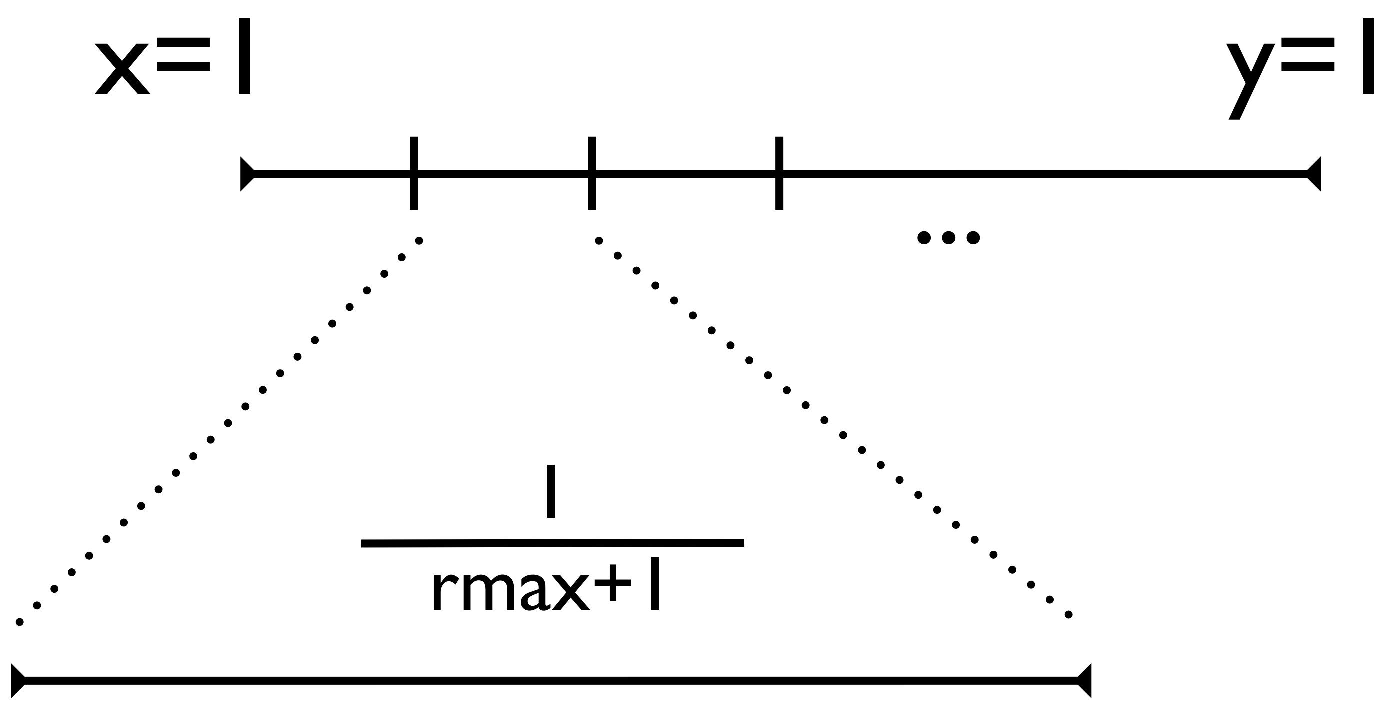
Proposition

- Let H be an RHA⊕ with a set of variables X
- Let ρ be a **T**-time bounded run of H
- Then ρ contains at most |X|•rmax•**T** transitions guarded by an equality

Proof:

- Use bounded time hypothesis
- False for transitions not guarded by an equality

Bounding between two equalities



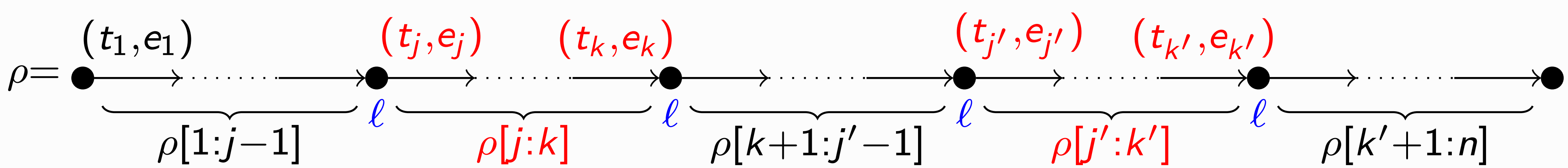
-no equality
-bounded time

→ shorten witness

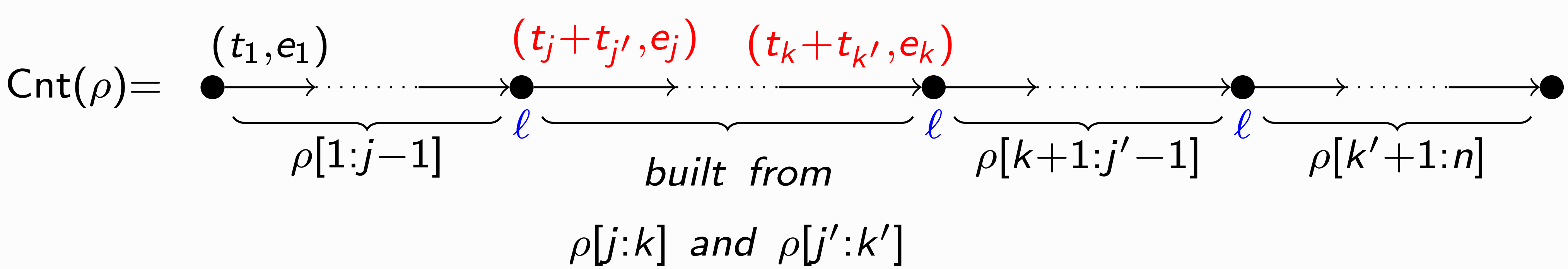
Bounding between two equalities

if big enough: cycles !

Key idea : The contraction operation

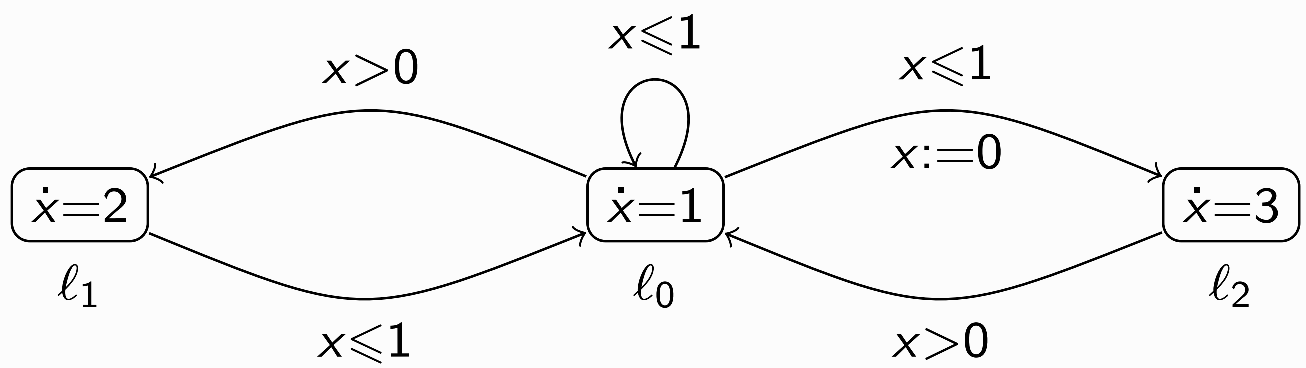


with $\rho[j:k]=\rho[j':k']$.



Bounding between two equalities

The contraction operation - A concrete example



$$\rho \;=\; (\ell_0,0) \xrightarrow{.1,e_{00}} (\ell_0,.1) \xrightarrow{.3,e_{01}} (\ell_1,.4) \xrightarrow{.1,e_{10}} (\ell_0,.6) \xrightarrow{.2,e_{00}} (\ell_0,.8) \xrightarrow{.1,e_{01}} (\ell_1,.9)$$

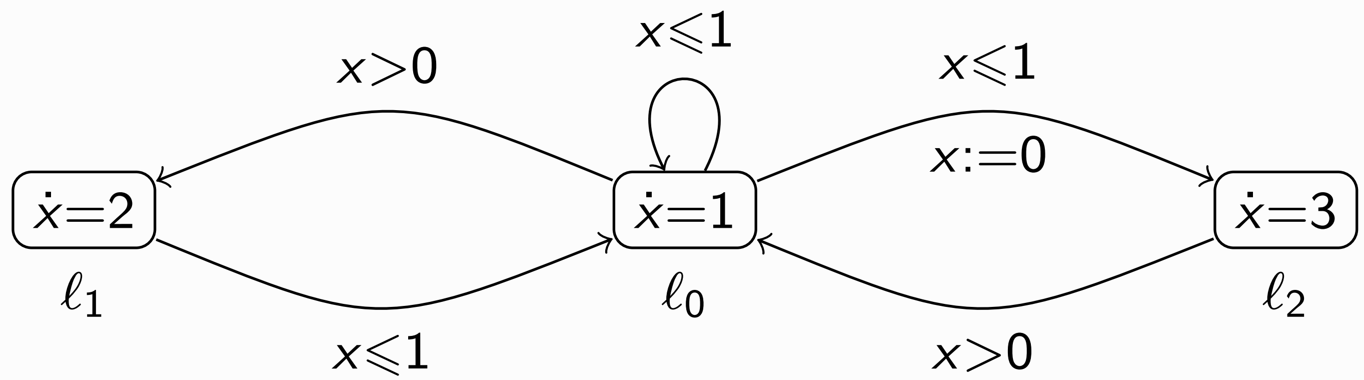
$$\mathsf{Cnt}(\rho) \;=\; (\ell_0,0) \xrightarrow{.1+.2,e_{00}} (\ell_0,.3) \xrightarrow{.3,e_{02}} (\ell_1,.6) \xrightarrow{.1,e_{20}} (\ell_0,.8) \xrightarrow{.1,e_{01}} (\ell_1,.9)$$

Advantages

- The new execution is shorter (in term of transitions).
- The value of the variables are preserved.

Bounding between two equalities

The contraction operation - Problem I



x:=0

$$\rho = (\ell_0,0) \xrightarrow{.1,e_{00}} (\ell_0,.1) \xrightarrow{.3,\mathbf{e_{02}}} (\ell_2,0) \xrightarrow{.1,e_{20}} (\ell_0,.3) \xrightarrow{.2,e_{00}} (\ell_0,.5) \xrightarrow{.4,e_{01}} (\ell_1,\mathbf{.9})$$

$$\mathsf{Cnt}(\rho) = (\ell_0,0) \xrightarrow{.1+.2,e_{00}} (\ell_0,.3) \xrightarrow{.3,e_{02}} (\ell_2,0) \xrightarrow{.1,e_{20}} (\ell_0,.3) \xrightarrow{.4,e_{01}} (\ell_1,\mathbf{.7})$$

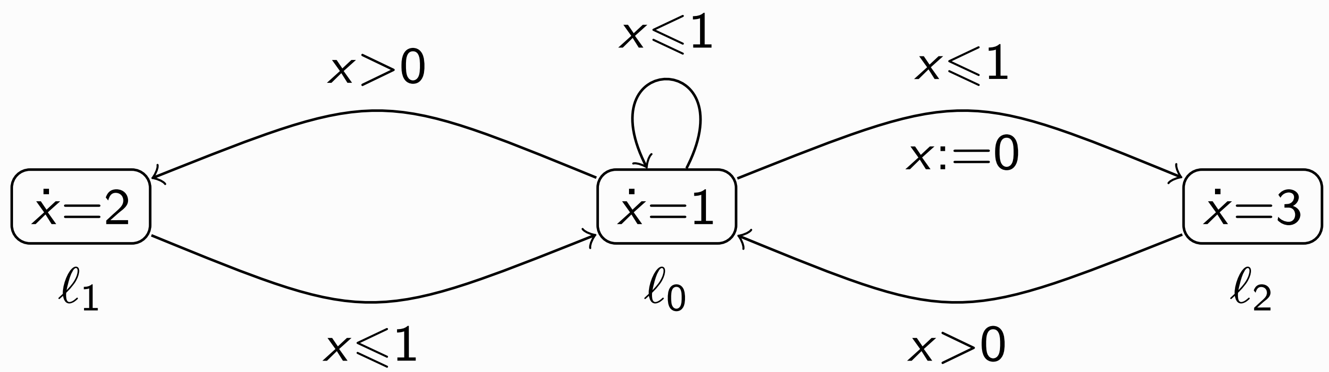
The value of the variables are not necessarily preserved...

Solution

Do not contract transitions occurring before and after the **last reset.**

Bounding between two equalities

The contraction operation - Problem II



x:=0

$$\rho = (\ell_0,0) \xrightarrow{.7,e_{00}} (\ell_0,.1) \xrightarrow{.3,e_{01}} (\ell_1,.4) \xrightarrow{.1,e_{10}} (\ell_0,.6) \xrightarrow{.2,e_{00}} (\ell_0,.8) \xrightarrow{.1,e_{01}} (\ell_1,.9)$$

$$\mathsf{Cnt}(\rho) = (\ell_0,0) \xrightarrow{.7+.2,e_{00}} (\ell_0,.9) \xrightarrow{.3,} (\ell_1,\mathbf{1.2})$$

Cnt (ρ) is not necessarily a proper execution...

Solution

- Do not contract transitions occurring before and after the **first reset**.
- Ensure that the time spent along an execution is **short enough**.

Bounding between two equalities

Building a bounded witness

Ultimate Goal

Given ρ an execution of $\mathcal{H}$ reaching ℓ_1 from (ℓ_0, x_0) within T time units.
We want to build ρ' such that :

- an execution of $\mathcal{H}$ reaching ℓ_1 from (ℓ_0, x_0) within T time units,
- the number of transitions of ρ' is bounded by a constant depending only of $\mathcal{H}$ and T.

- **Step 1 : Time-slicing**
  We can slice ρ is pieces whose duration is at most $\frac{1}{R_{max}}$.

  At most $R_{max} \cdot T$ pieces.

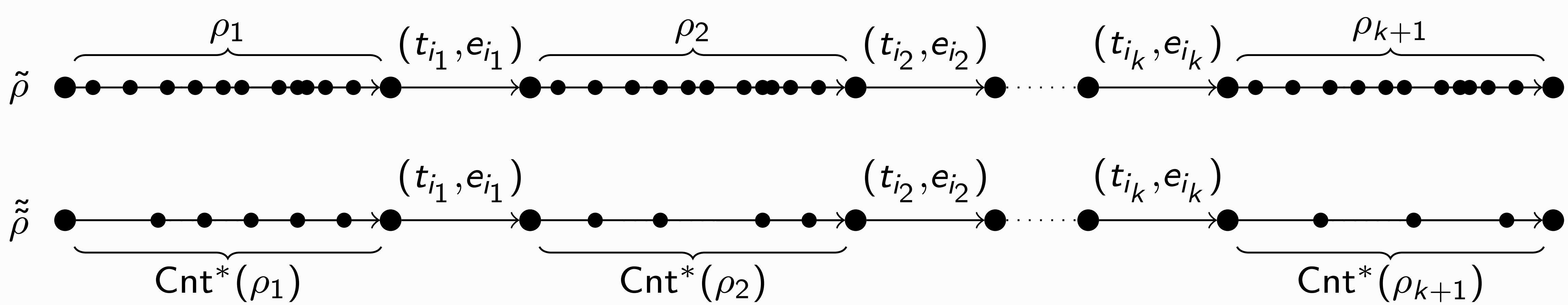
- **Step 2 : First and Last reset-slicing**
  We can slice ρ according to the first an last resets of each clock.

  At most $3 \cdot |X|$ pieces.

Bounding between two equalities

Building a bounded witness (continued)

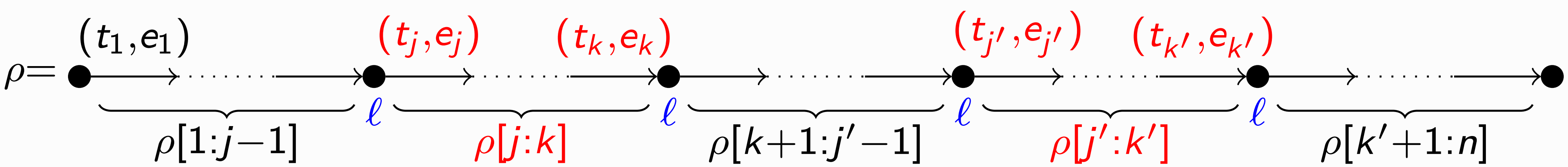
- **Step 3 : Application of the contraction :**



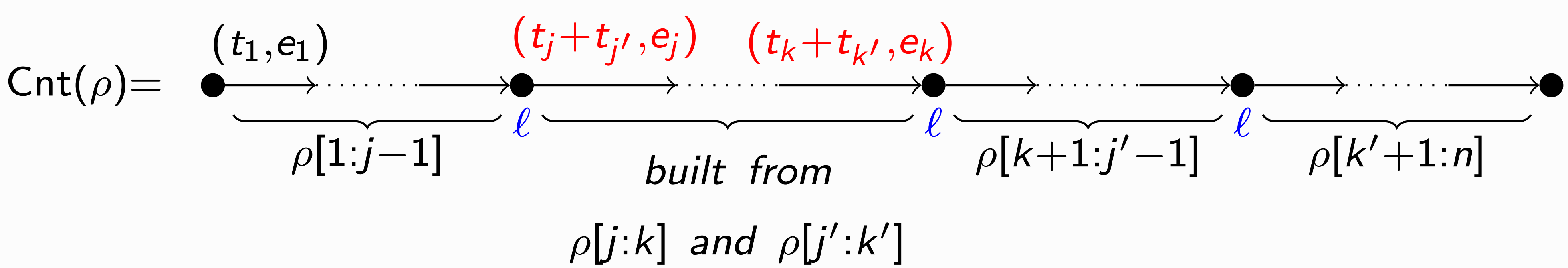
- $\tilde{\tilde{\rho}}$ is a proper execution of $\mathcal{H}$.
- The variables have the same value at the end of $\tilde{\rho}$ and $\tilde{\tilde{\rho}}$.
- The number of transitions in $\tilde{\tilde{\rho}}$ is bounded by a constant depending only of $\mathcal{H}$.

Bounding between two equalities

The contraction operation



with $\rho[j:k] = \rho[j':k']$.



$$|\mathsf{Cnt}^*(\rho)| \leq |\mathrm{Loc}| \cdot (2^{(|\mathrm{Edges}|+1)} + 1),$$

where $\mathsf{Cnt}^*(\rho)$ is the fixed point obtained by iterating $\mathsf{Cnt}(\cdot)$ to ρ.

Decision procedure for TBR

Theorem

A **Goal** location is reachable in RHA⊕ H within **T** time units

iff

it is reachable by a run ρ of **size bounded by K**(H,**T**)$\in\mathbb{N}$.

Corollary

Time bounded reachability can be reduced to the **satisfiability** of a **formula in the first order theory of the reals** encoding the existence of runs of length at most **K**(H,**T**) that reaches Goal.

Decision procedure for TBR

Theorem

A **Goal** location is reachable in RHA⊕ H within **T** time units

iff

it is reachable by a run ρ of **size bounded by K**(H,**T**)$\in\mathbb{N}$.

Corollary

Time bounded reachability can be re[redacted]
formula in the first or[redacted]
existence of [redacted]al.

Not based on a finite similarity quotient
(as it is usually the case)

Beyond RHA⊕

- Negative rates lead to **undecidability**
- Diagonal constraints lead to **undecidability**

Decidability frontier

	Reach	Time-bounded Reach
Timed automata	☝	☝
Initialized RHA	☝	☝
RHA⊕	☟ (Stopwatch)	☝
RHA	☟	☟ (neg. rates or diag.)
LHA	☟	☟
Affine HA	☟	☟
O-Minimal HA	☝	?

Conclusion

- Reachability analysis of hybrid automata have proven **useful** (embedded systems-protocols-biological systems-etc.)
- **PhaVer** and **HyTech** implements symbolic semi-algorithm for LHA-RHA
- PhaVer implements rectangular approximations of affine HA

  Details: Laurent Doyen, Tom Henzinger, Jean-François Raskin. **Automatic Rectangular Refinement of Affine Hybrid Systems**. In FORMATS'05, Lecture Notes in Computer Science 3829, pp. 144--161, Springer-Verlag, 2005.
- **Time-bounded** reachability is **decidable** for RHA⊕ (⊇stopwatch HA)

  Details: Thomas Brihaye, Gilles Geeraerts, Laurent Doyen, Joel Ouaknine, Jean-François Raskin and James Worrell. **On reachability for Hybrid Automata over Bounded Time**. In ICALP'11, LNCS 6756, Springer, pp. 416-427, 2011.